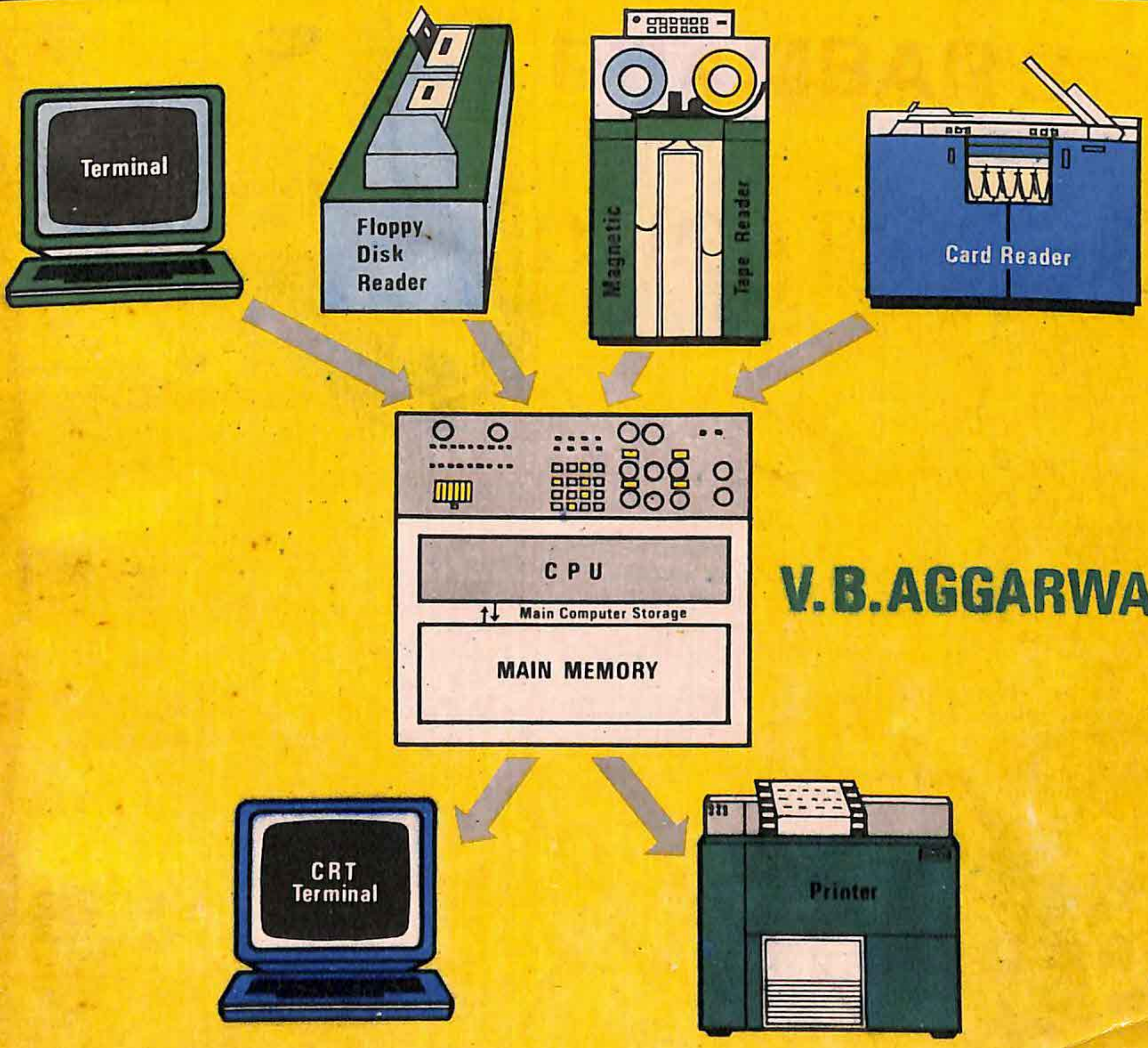
A NEW CERTIFICATE COMPUTER STUDIES



V. B. AGGARWAL

PITAMBAR PUBLISHING COMPANY

Based on the latest syllabus prescribed by The Council For the Indian School Certificate Examinations, New Delhi in Computer Studies For ICSE

A NEW CERTIFICATE COMPUTER STUDIES

[For Classes IX-X]

BY

V.B. AGGARWAL M.S., Ph.D. (U.S.A.)
Founder-Head,
Department of Computer Science
University of Delhi,
and Ex-Professor of Computer Science,
University of Oklahoma, U.S.A.

S.C.E.R.T., W.B.
Library
Calcutta
25.3.

PITAMBAR PUBLISHING COMPANY
888, East Park Road, Karol Bagh,
New Delhi-110005
(INDIA)

Our Prestigious Publications on Computer Science

1. **Workbook in BASIC** *by V. B. Aggarwal & M. P. Goel*
2. **Programming in PASCAL** *by N. L. Sarda*
3. **Microcomputer: Programming & Utilization** *by N. L. Sarda*
4. **COBOL programming with Business Applications** *by N. L. Sarda*
5. **Programming in BASIC** *by N. L. Sarda*

Published by :
PITAMBAR PUBLISHING COMPANY
888, East Park Road, Karol Bagh,
New Delhi-110005 (INDIA)
Telephones : Off. : 770067, 776058, 526933
Res. : 5715182, 586788, 5721321

© Copyright with the Author

S.C.E.R.T. W.B. LIBRARY
Date 5. 11. 98
Accn. No. 9758

First Edition : 1989
Reprinted : 1989
Second Edition : 1990

Code No. 27209

All Rights Reserved
(No part of this book including diagrams may be reproduced by any means nor transmitted or translated into any language, without the written permission of the author/publisher.)

Price : Rs. 22/-

Printed at :
PIYUSH PRINTERS PUBLISHERS (P) LTD.,
G-12, Udyog Nagar,
Rohtak Road Industrial Area,
New Delhi-110041
(Phone : 5472440)

PREFACE

In this computer-conscious environment, there is a great need for a book that introduces the basic concepts in a simple and lucid manner. As common man is going to interact with computer systems in his daily life, he must not be afraid of computers and similar machines. This book dispels any wrong notions about the use of computers. In this book computer fundamentals have been introduced so as to make it easy to learn and use computers. The other important aspect of this book is to introduce the discipline of problem-solving in a systematic manner. It gives a thorough presentation of problem-solving and flow-charts for computers.

The Council for Indian School Certificate Examinations has taken a very progressive step in introducing computer studies as one of the six subjects in the examination scheme of Indian Certificate of Secondary Education. Thus the students of IX & X classes shall have an opportunity to study the latest and most innovative science of our times. Computer technology has already made a lasting impression on our life. It shall certainly further influence our life-style and working environment. The new generation must be well-prepared for these technological challenges which are already visible on the horizon. Therefore, it has been indeed a pleasure to write this book to meet the needs of young and bright students appearing in the ICSE examination.

This book fully covers the ICSE syllabus on Computer Studies for IX & X classes. The computer material has been covered systematically in eleven chapters of the book. The matter is presented in a graded manner and it encourages the students to learn the fundamentals. Exercises, questions and problems are given at the end of each chapter. Working through these would clarify the concepts covered in each chapter and generate confidence in the students. Some of the exercises are suitable for writing computer programs which can be then tried on the computer during the practicals. The book thus meets the theory and practical requirements of the ICSE syllabus.

In addition, the book can be used effectively by other people who want to get started on computer studies. I would consider my efforts fruitful if this book succeeds in arousing interest in computers and develop logical thinking among the readers. I would like to acknowledge the assistance given by the technical staff of Computel Systems & Services which has been very helpful in bringing the manuscript to its final shape. I would welcome any comments and suggestions for the improvement of the book from the readers.

— Prof. V.B. Aggarwal

SYLLABUS IN COMPUTER STUDIES

PRESCRIBED BY
COUNCIL FOR THE INDIAN SCHOOL CERTIFICATE EXAMINATION
FOR INDIAN CERTIFICATE OF SECONDARY EDUCATION

Introduction :

(i) Computer Studies is an applied subject and the aim of this syllabus is to reinforce the practical aspect by dealing with the following questions:

1. How does a computer work ?
2. How is information stored ?
3. How do you get information into it ?
4. How do you get information out of it ?
5. How do I make it work for me ?
6. What do people do with computers ?

(ii) The syllabus is based on a teaching schedule of 3 periods per week for 30 weeks in each of the two years, Class IX and Class X; at least two thirds of this time should be used for programming and real time experience.

Part 1. *There will be* one *paper of 2 hours.*
The paper will be divided into two Sections A and B.
Section A will consist of questions requiring only short answers and will cover the whole syllabus. There will be no choice of questions.
Section B will consist of traditional type questions. There will be a choice of questions and candidates will be required to answer **four** *questions.*

Note: The syllabus is intended to indicate the scope of the course and not to dictate the teacher's approach to the subject.

SYLLABUS	NOTES
1. History of Computers	Abacus, calculating devices, processing devices, early computers, various generations of computers.
2. Binary State	Base two, base eight, base sixteen arithmetic, as required by a computer: bits, bytes, word ; role of the chip, materials used for chips.
3. Structure of the Computer (including information retrieval)	
(*a*) *Input Devices*	Punched cards, console keyboards, bar codes, magnetic ink character readers. (MICR)

(*b*) *Central Processing Unit (CPU)*	Memory or immediate access store or working store, arithmetic/logic unit, control unit.
(*c*) *Output Devices*	Visual display unit (VDU), including graphs, line printer, daisy-wheel printer, card punch.
(*d*) *Auxiliary storage*	Magnetic tapes, magnetic discs, optical disc, memory.

Notes: (i) Candidates should know the location of each unit in a micro-computer and where possible a mini-computer and main-frame ;

(ii) diagrams, models, photographs and other visual aids may be used.

4. Flowcharts and simple programs	Symbols used in flowcharting relevant to a four-line program ; learning to make corrections in the program.
5. Computer Logic	Boolean logic, truth tables, logic gates ; logic circuit.
6. System Software	Computer language, flowcharting and writing programs in BASIC upto twelve lines.
7. Data Processing	Distinction between information and data ; digital and analog computer (diagrams, photographs and visits) ; input of data ; how the computer deals with data ; output of data ; data storage ; batch processing; on-line procesing, real-time processing.

Part 2. *To be assessed internally by the school*

Practical or Project Work in Computer Studies
Use of computers and their applications:

(*a*) Word processing :

(*b*) Application of computers to :
Learning and instruction.
OR banks
OR household appliances,
OR one commercial organisation,
OR an industry,
OR using laboratory data to verify a law or principle in science *e.g.*, Newton's laws of motion, cooling curves for various substances,
OR other suitable projects.

* * *

CONTENTS

Chapter No.		Page No.
1	History of Computers	1
2	Binary State	15
3	Computers	25
4	Structure of the Computer	29
5	Input/Output Devices	35
6	Problem Solving	50
7	Flow Chart and Simple Four-line Programs	54
8	Computer Logic	80
9	Introduction to BASIC	97
10	System Software	111
11	Data Processing	123

1

HISTORY OF COMPUTERS

The discovery of numbers, which is considered as the greatest achievement of mankind, led to the development of all kinds of calculating devices. Today we cannot imagine a world without numbers. It took thousands of years to get the present system of numbers. Its development passed through several stages.

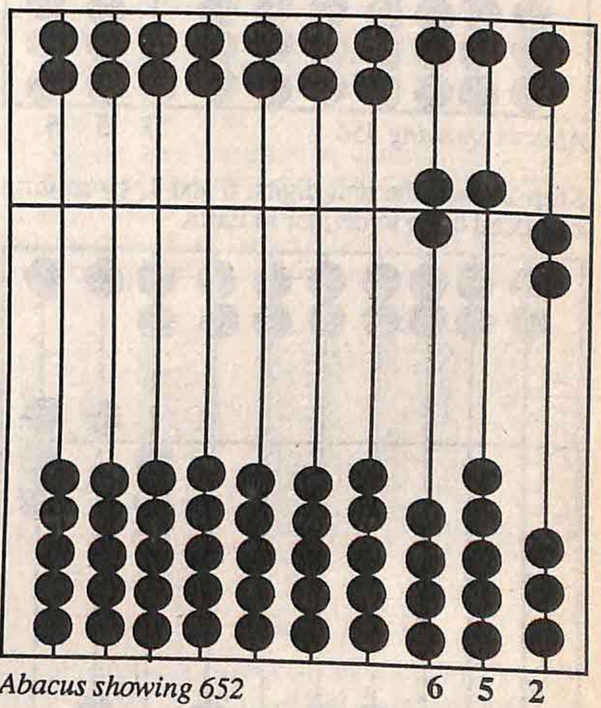
The stone age man used pebbles for counting people or animals. When sheep left its fold the herdsman put one stone in his pocket, and he continued doing so, till all sheep left the fold. When sheep returned to the fold, he repeated the above procedure in reverse order by taking out a stone from his pocket for each sheep. Finally if his pocket contained no stone then only he was sure that all the sheep had come back. Thus he laid foundation for a very important concept of modern mathematics, one-to-one correspondence. (one stone for one sheep).

Stones were replaced by sticks, scratches, symbols on stones and knots on a rope. All these computing techniques required some physical effort from man. A lot of work was required for large calculations and the process was very slow. Later these scratches and symbols were replaced by the present number system. Number system found its use in astronomy, building temples & churches and calculating areas. The discovery of zero by Indian mathematicians laid the foundation stone for numeric system. Let us now study some important developments enroute to the invention of modern day computers.

THE ABACUS

The earliest and the simplest device that was used for calculations was the Abacus. Simple Abacus has been in use since 350 B.C. Abacus was first developed in China and later its use spread to all parts of the world. Even today, it is still in use in many countries like Soviet Union, Japan, China and other countries in Far-East. It was the first device to introduce the concept of **Data storage.**

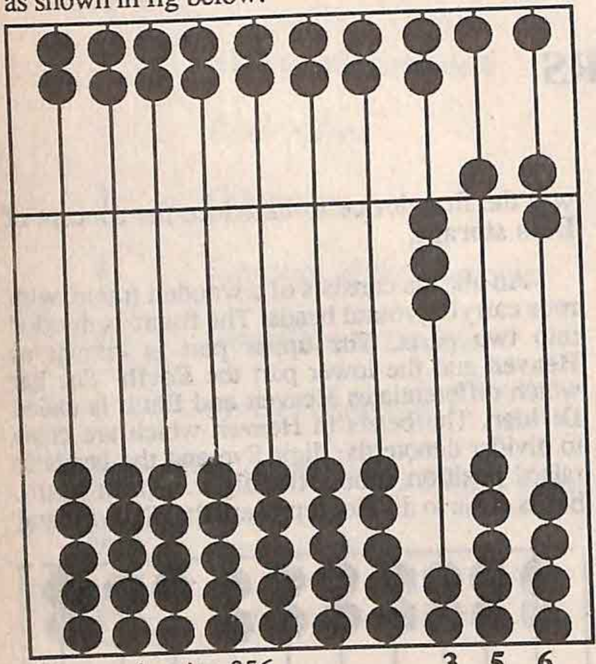
An abacus consists of a wooden frame, with rods carrying round beads. The frame is divided into two parts. The upper part is known as **Heaven** and the lower part the **Earth**. The bar which differentiates Heaven and Earth is called **Divider.** The beads in Heaven which are close to divider denote the digit five and the beads in raised position denote the digit zero. In Earth, beads close to divider represent the digit one and the ones in lower part represent zero. In general, each rod in Heaven of an Abacus has two beads and in Earth five. The above figure illustrates the method of representation of a number on an abacus.



Abacus showing 652

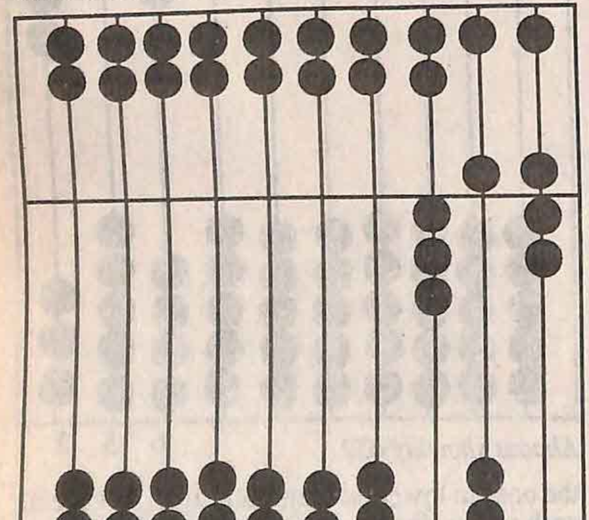
To understand the principle of abacus, let us consider a simple example of addition of two numbers say 356 and 861.

Step 1 : Set up the number 356 on the abacus as shown in fig below:



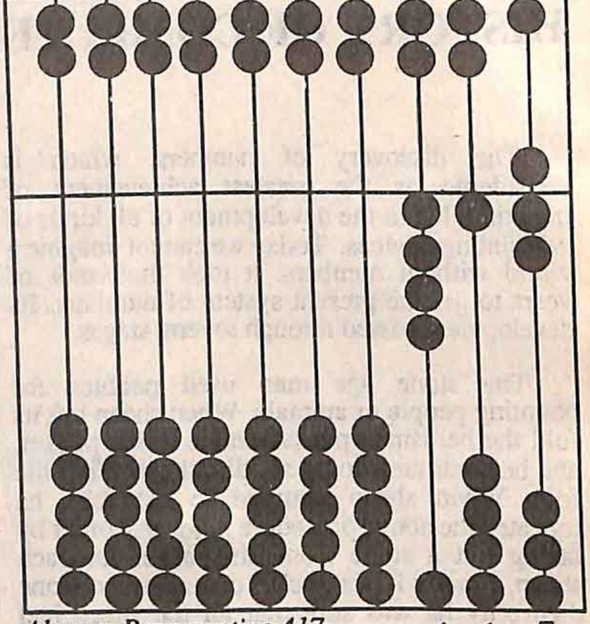
Abacus showing 356

Step 2 : Add the unit digits, 6 and 1, by shifting one bead close to divider in Earth.

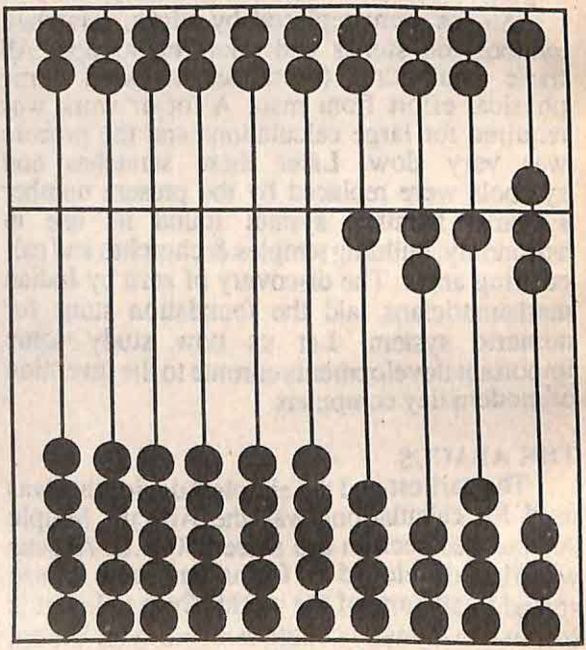


Abacus showing 357

Step 3 : Add the tens digits: 5 and 6. This requires shifting of 6 beads in Earth close to divider. But there are only 5 beads in Earth. As we are short of one bead, move one bead in Earth close to divider and move all the beads in Heaven to the top. Now move one bead in hundreds position close to divider.



Abacus Representing 417



Abacus Representing 1217

Step 4 : Add the hundred's digits 4 and 8. This requires shifting of 8 beads close to divider. But there is only one bead in Earth. So we can move 6 close to divider by moving one bead from Heaven and one from Earth. As we are short of two beads, we keep two beads close to divider and move one bead of thousands in Earth close to divider.

The operations: multiplication and division are done using repeated additions and subtractions.

Though it looks very slow, in the hands of an experienced person the abacus can compete with the best mechanical desk calculator.

NAPIER'S 'LOGS' and 'BONES'

John Napier (1550-1617), a Scottish mathematician invented logarithms. The use of logarithms enabled him to transform multiplication and division problems into problems of addition and subtraction. In the beginning he called logarithms as artificial numbers. But later he named them logarithms. It is a combination of two Greek words, **logos,** (meaning ratio) and **arithmos** (means number).

When he first invented these numbers, he had no idea of base. Later in 1624, Professor Henry Briggs of Gresham College, London, defined base and tabulated logarithms to the base e (e is a number whose value is app. 2.71) and base 10.

The invention of logarithms was an important breakthrough in the development of mathematics and computing devices.

We may define a logarithm as an exponent. For example, in the equation $3^3 = 27$, the number 3 is an exponent which is applied on the base 3 to give the number 27. In general consider the equation $a^x = N$. Here, x is an exponent which when applied to the base a will yield the number N. Then x is defined as the logarithm of N to the base a. Hence the logarithms of a number to a given base is the index of the power to which the given base must be raised to get that given number.

To ease the calculation work using logarithms the values of logarithms are immediately available from a book known as **logarithm tables.**

Logarithms to the base 'e', are called **Natural logarithms** and are widely used in theoretical calculations. The logarithms to the base 10 are called **Common logarithms.** Common logarithms are widely used in all practical calculations. The following properties of logarithms are useful in using logarithms for computing expressions:

i. The logarithm of the product of two numbers is equal to the sum of the logarithms of the individual numbers (to the same base).

$$\log m.n = \log m + \log n$$

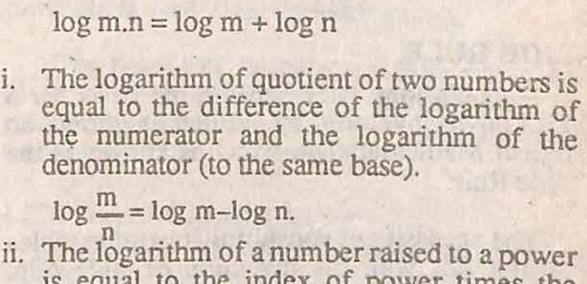
ii. The logarithm of quotient of two numbers is equal to the difference of the logarithm of the numerator and the logarithm of the denominator (to the same base).

$$\log \frac{m}{n} = \log m - \log n.$$

iii. The logarithm of a number raised to a power is equal to the index of power times the logarithm of the number to the same base.

$$\log m^n = n \log m$$

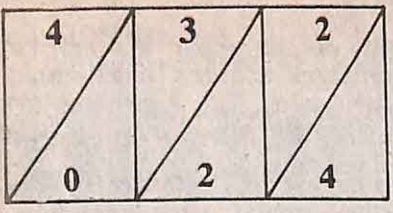
Napier also invented a computing device consisting of eleven sticks with numbers carved on them. These sticks are also called "bones" as they were made of bones. These bones helped a lot in multiplication and division involving large numbers.



Napier's Bones

Suppose one wants to multiply 543 with 8. Then the bones with 5, 4 and 3 are arranged as shown below and then consider the 8th row.

Now copy the right most digit to units place of the result i.e., 4. Now add the numbers diagonally to get the digits in tens, hundreds, thousands places and copy the last digit in the



Napier's Bone Showing 543

upper most corner to get the most significant digit of the result.

2 + 2 = 4 ——— Ten place.
3 + 3 = 3 —— hundred place.
4344

SLIDE RULE

Napier's logs later became the basis for a well known invention by Willian Oughtred, an English Mathematician, which is known as the "Slide Rule".

The necessity of consulting logarithm tables was not felt with the invention of slide rule, where the final value can be straight away read on the scale.

The essential parts of a slide rule are:

(i) The body of the rule, which carries scales. This part is stationary.

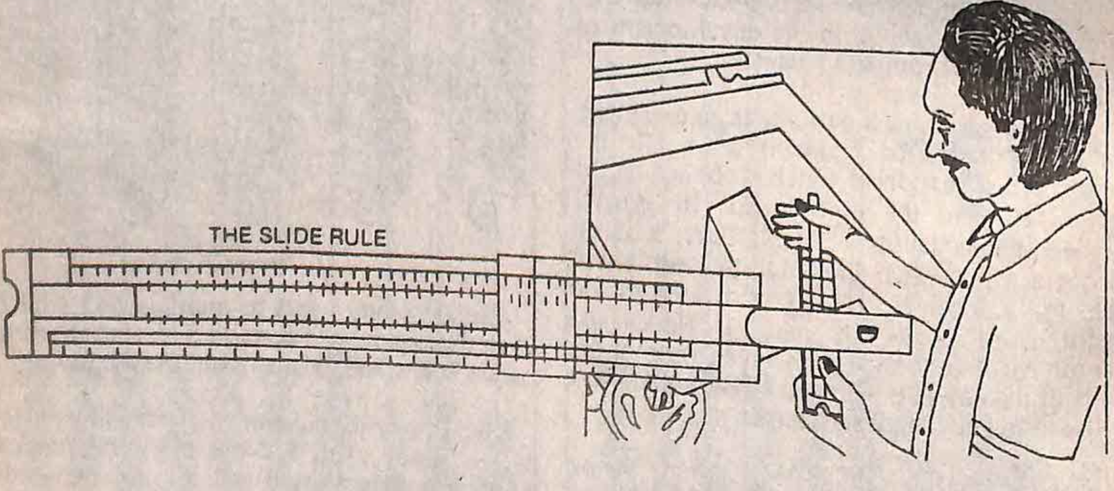
(ii) The slide carries scales and is movable; and

(iii) The cursor or hairline indicator.

The slide operates in a slot in the body of the rule and it can be moved right or left. The cursor is made of glass or plastic with a thin line on its lower surface, perpendicular to the edges of the slide. This line helps in reading the scales, to indicate the initial, intermediate and final settings, to match the readings on different scales and to indicate the final result of computation. A simple slide rule consists of four scales called A, B, C and D. Two of these scales are on the body of the rule and the other two are on the slide.

As these scales are logarithmic, the spacings are not uniform but are proportional to the logarithms of the numbers. Scales C and D are exactly alike and their both ends start with number 1 which is called index. Between these indices, each scale is divided into appropriate subdivisions. The distances of these subdivision marks, from the left index are proportional to the logarithms of the numbers associated with the subdivision marks. The A and B scales are also logarithmic and exactly same. The actual spacings on these scales are double the corresponding spacings on the D scale.

Now let us consider the problem of



Slide Rule

multiplication of one number by another number. Multiplication is usually performed on the C and D scales. But it can also be performed on A and B scales. The procedure, using C and D scales, is as follows :

(i) Place the line of the cursor on the first number on the D scale.
(ii) Place the left index of the C scale (which is on slide) under the line of the cursor by moving the slide.
(iii) Now move the cursor and place the line of the cursor on the second number on C scale. Do not move the slide when the cursor is moving.
(iv) Now read the number, which is the product of the two numbers, on the D scale which is just under the line of the cursor.

There are many varieties of slide rules used as calculators. With a simple slide rule one can perform multiplication, division, find squares, square roots, proportions, percentages and also solve linear equations.

The principle used in Slide Rule is that the actual distances from the starting point of the rule are proportional to the logarithms of the numbers printed on the rule. The multiplications and divisions are done by moving the transparent rectangular piece.

PASCAL'S ADDING MACHINE

The next attempt in the computing history was made by the great French Mathematician Blaise Pascal. At the age of 18, in the year 1642, to help his father in computations, Pascal invented a machine which used gears, cog wheels and dials. This machine displayed numbers by the rotation of cog wheel. This machine was used only for addition and subtraction of numbers. Subtraction was done by using complement of numbers. The same principle is used in modern computers.

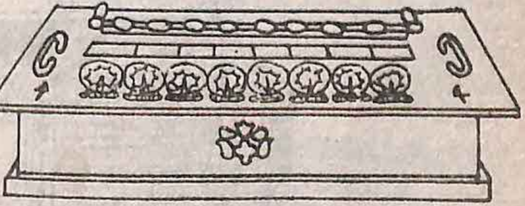
The design of Pascal's machine was ahead of its time. It could not be produced by the technology of the time as there was no industrial technology in the 17th century.

LEIBNITZ'S CALCULATOR

Gottfried Van Leibnitz, a famous German Mathematician, improved on Pascal's machine in 1671. His machine was able to perform all the four basic mathematical operations : Addition, Subtraction, Multiplication and Division.

Leibnitz's machine used stepped cylinders, each with nine teeth of varying length, instead of

Blaise Pascal



Pascal's Arithmetic Machine

wheels used by Pascal. He introduced a new concept—**shift mechanism.** Moving a digit to right or left is called shift. He achieved this through a series of sliders. But Leibnitz's calculator was also too advanced for the technology of its times.

The draw back of the above machines was that they were very slow and needed human intervention at every stage. Also they were quite unreliable and lacked in accuracy. So efforts were made to design fast, accurate, reliable and automatic machines.

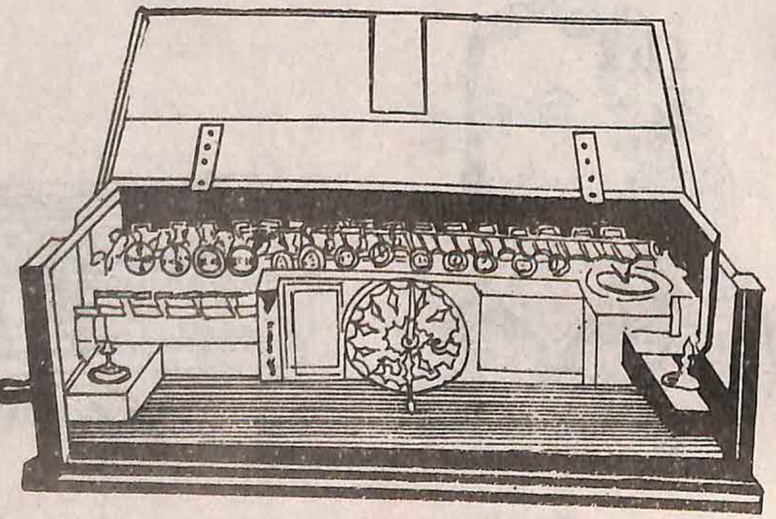
JACQUARD'S LOOM

Joseph Jacquard initiated actual developments in the early nineteenth centry. Jacquard was a textile manufacturer. He invented **Automated Loom** in 1804. He used punched cards to produce complex cloth patterns with Automated Loom. The function of the loom depended upon the absence or presence of holes on the card. Thus Jacquard's loom paved the way for the modern storage mechanism on punched cards, and the binary coding system.

BABBAGE'S ANALYTICAL ENGINE

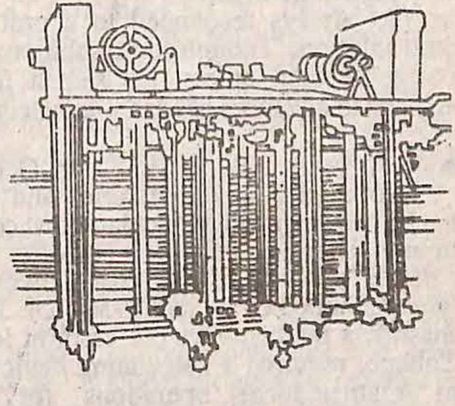
Charles Babbage, a British Mathematics Professor, known as the Father of Computers developed and demonstrated a working model of a mechanical computer called **Difference Engine** in 1822. This machine was based on the principle that the difference between certain values of the expression at a certain stage becomes constant. Encouraged by the success of Difference Engine, he came up with another device called **Analytical Engine** in 1833. Analytical Engine used the concepts of Automated Loom and Difference Engine. This machine had five units — Input, Output, Store, Mill and Control, which are similar to the units of a modern computer. Store was used for storing the numbers and Mill used to do all arithmetic operations by rotation of gears and wheels. Control unit did the job of supervision of all other units. The input material in the form of Programs, was developed by Lady Ada Lovelace. Due to this contribution, a language called Ada was named after her.

Among the many innovations incorporated in the analytical engine, two were revolutionary. The first enabled the machine to compare quantities and then decide which of the instruction sequences to follow, the second permitted the results of a calculation to change numbers and instructions already stored in the machine.



Leibnitz's Calculator

Charles Babbage



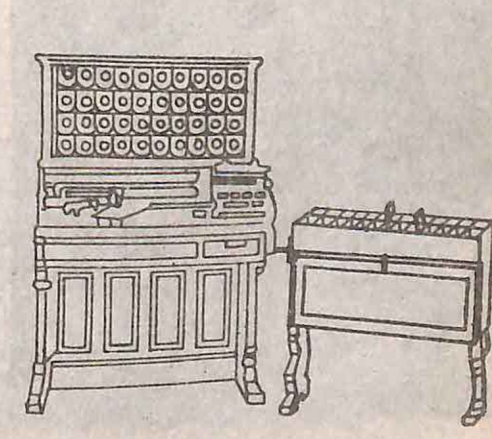
Babbage's Analytical Engine

Due to lack of technology of that time, the Analytical Engine was never completed.

HOLLERITH'S MACHINE

At the end of the 19th century, the analysis of data relating to census in U.S.A. became very difficult for the Census Bureau. Then, in 1889, Herman Hollerith, a census statistician with U.S. Census Bureau, devised a machine run on electricity for the first time, for calculations and recording data. Hollerith's machine was capable of reading both numbers and letters, processing them and giving out letters and numbers in the desired form. The input was in the form of punched cards and the combination of holes formed the 'code' of input data. He invented a machine that punched holes on cards, which was called **Tabulator**. This machine processed the cards by detecting holes by electric contacts and also performed the function of selecting and sorting of the cards. This machine solved the information of 60 million people of 1890 U.S.

Dr. Herman Hollerith



Hollerith's Tabulating Machine

census just in three years. In 1896, Hollerith established his own company called Tabulating Machine Company (TMC). The same company is today known as International Business Machines Corporation (IBM).

LATER DEVELOPMENTS

Man has always responded to a problem with some solution. Though some solutions are less acceptable than others, the search for a solution leads to advances in Man's knowledge.

About 50 years ago, in 1937, Dr. George Stibitz approached Bell Laboratories and told them that he had designed a Calculator that could perform any general calculation. His offer was turned down because of the cost involved which was $50,000.00. In late 1930's Dr. John V. Atanasoff, a mathematics professor at Iowa State College, required a calculating device to perform mathematical operations for his students. Atanasoff, after examining the devices then available, concluded that they were inadequate for his needs and that the solution to his problem lay in the development of a digital computing device. Atanasoff set about designing his own machine. With the help of his assistant Berry he designed the Atanasoff Berry-Computer (ABC). This machine laid the foundation for the development of electronic digital computer.

ABC Computer

Inspired by ABC, John W. Mauchly met Atanasoff in 1940. At the same time he also got acquainted with J.P. Eckert JR.

During World War II, the U.S. Army had a need for calculating ballistic tables to produce trajectories for bombing. The army was using differential analyzers to calculate the tables. These mechanical devices could compute a 60 sec. trajectory in 15 min. Considering the demand for these tables 15 min. was very slow. In April 1943 Mauchly and Eckert submitted a report to U.S. Army describing an Electronic Difference Analyzer. This machine was able to do the above calculations in 30 sec. The real aspect of the idea was the proposal to build the machine using vacuum tubes. One mathematician commented that since the average life of a vacuum tube is 3,000 hrs, a tube failure would occur every 15 min. Since it would take more than 15 min. on an average, to find a bad tube, no useful work could ever be done. Despite of it in 1943, the army funded Mauchly and Eckert to begin the development of a new machine.

ENIAC (Electronic Numerical Integrator and Computer)

In 1946, Mauchly and Eckert completed the first large scale electronic digital computer ENIAC. After it had been in operation, Lord Mountbatten, President of the British Institute of Radio Engineers said, "The stage has now been set for the most wellsian development of an all electronic brain."

The ENIAC was programmed by connecting wires between units of the computer and setting switches in such a way that the program could be executed. Each time a program was changed, the wiring had to be completely redone. It weighed 30 tons, contained 18,000 vacuum tubes and occupied a space of 30 feet by 50 feet. It was used for preparing ballistic tables, weather prediction, atomic energy calculations and random number studies. On October 2, 1955, ENIAC was turned off for the last time.

Before the completion of the ENIAC U.S. Army asked Moore School of Electrical Engg. to build a computer more powerful than ENIAC in 1944. In response to this request, Dr. John Von Neumann described the EDVAC (Electronic Discrete Variable Automatic Computer) in his report submitted to U.S Ordinance Department.

In his report he suggested the **"Stored Program Concept"** i.e., to place the computer instructions in computer's main memory in the form of numbers in a manner similar to the way data is stored for processing. Whenever a new program was to be executed, the program would be read into the memory instead of changing switches and wires.

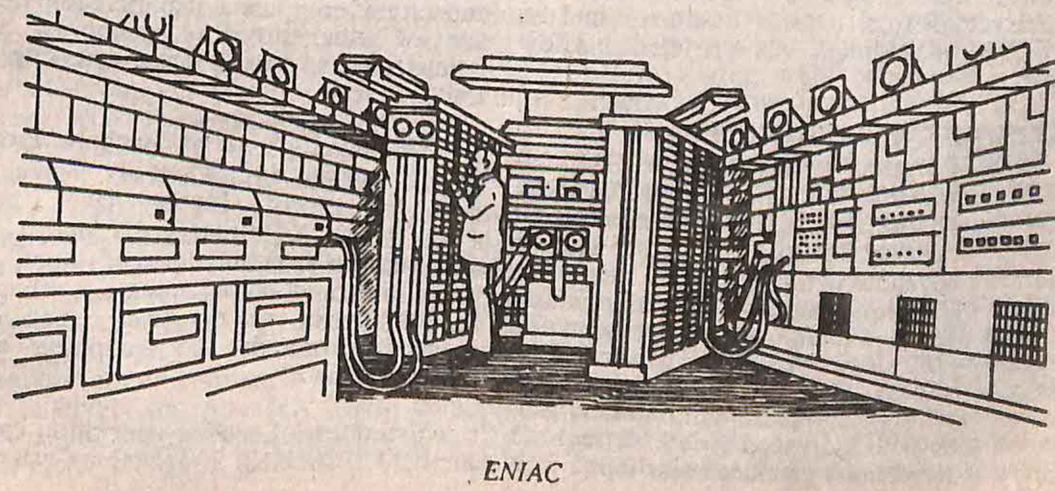
Inspired by the lectures of Von Nuemann and others at University of Pennsylvania, Maurice V. Wilkes of Cambridge University alongwith his colleagues at Cambridge began work on EDSAC (Electronic Delay Storage Automatic Calculator) and subsequently completed it in 1949. EDSAC was the first computer to operate on the stored program concept. Although EDVAC was proposed in 1945 with stored program concept, it started operating only in 1951. ENIAC, EDVAC, EDSAC and other computers which were developed in late 1940's were on experimental basis. In 1947, after ENIAC became operational, Mauchly and Eckert formed their own Company - The Eckert - Mauchly Computer Corporation. Immediately after this they started the design of UNIVAC I (Universal Automatic Computers). This machine was purchased by U.S. Bureau of Census for use in the 1950 census. UNIVAC was the first computer dedicated to business applications. It was also used for the analysis of 1952 Presidential Elections in United States.

GENERATIONS OF COMPUTERS

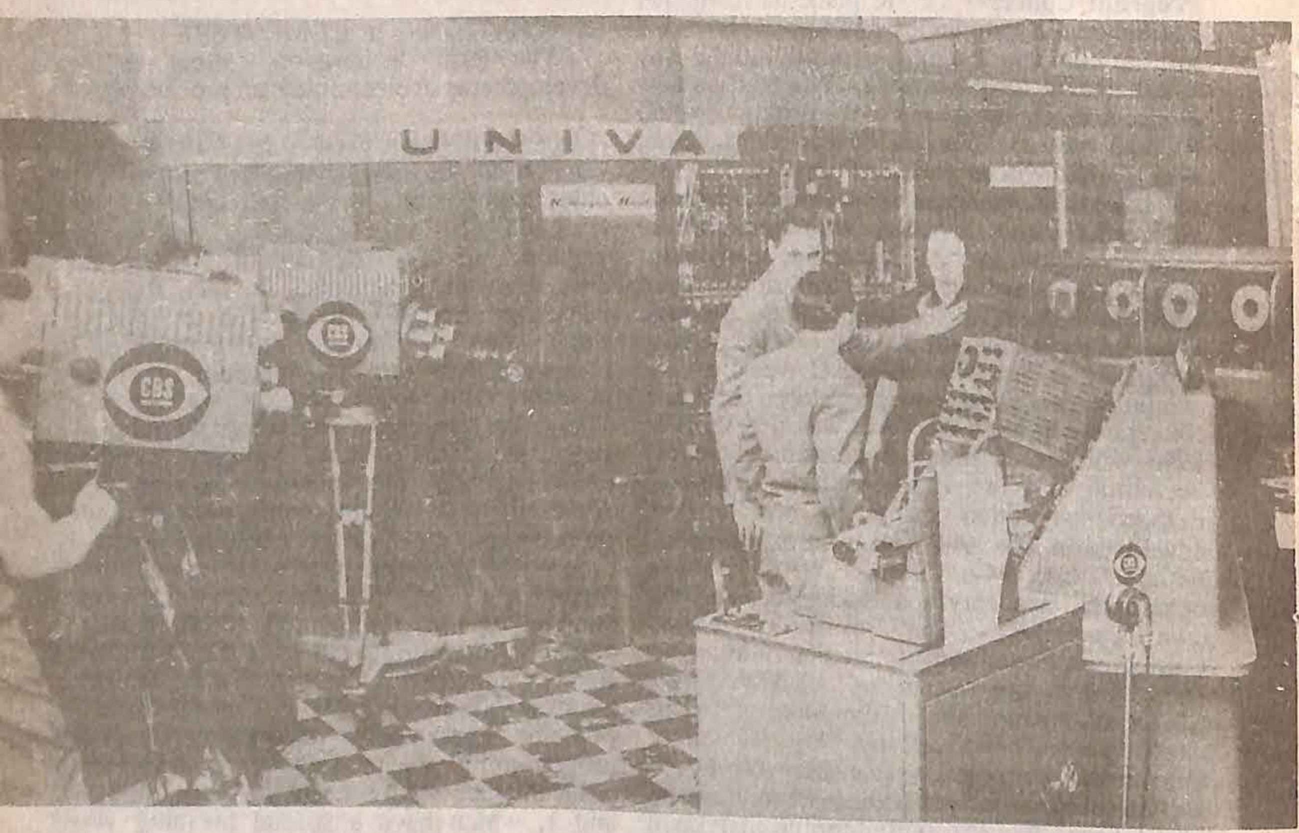
The term "generation" refers to major developments in electronic data processing.

FIRST GENERATION (1945-1959)

In all the first generation computers the chief components were (ENIAC, EDVAC, EDSAC, UNIVAC, etc.) **valves and vacuum tubes.** The use of vacuum tubes created several problems. In fact, the power needed by ENIAC was enough to light up a small town. To protect other units of the computer from the heat, vacuum tubes were insulated and water cooling was installed. These first generation computers were large in size, unreliable and required constant maintenance. Storage technique was primitive. Although the stored programme concept made a tremendous impact on programming, the programming process was cumbersome, error-prone and difficult. In first generation computers, the programmer who wrote a computer program had to write the instructions for the computer in a language that the computer hardware understood. This language is called Machine Language. In machine language programming, the programmer is required to write instructions by means of binary digits, 0 and 1, which have a special meaning when



ENIAC



The UNIVAC – I Computer

understood by the computer. It was very difficult to write programs in machine language. Moreover most programs contained errors. It became very difficult to trace these errors and the knowledge of language was restricted to a few people.

SECOND GENERATION (1959-65)

Three scientists J. Bardeen, H. W. Brattain and W. Shockley, for which they received Nobel Prize later, invented the **Transistor** at Bell Laboratories in 1947. This led to significant advances in computers. A transistor is made of a semi-conductor material whose principal part is silicon. Vacuum tubes were replaced by transistors in 1959. The size of the transistor is only 1/200th of the size of the vacuum tube. It performs the same functions as vacuum tube with increased speed. It requires much less power and produces less heat. It is cheaper to produce and operate and far more reliable. Computers using transistors were called Second Generation Computers. Storage capacity of computers also increased due to the use of magnetic core memories. These computers had inbuilt error detecting devices also.

The development of Symbolic Programming languages i.e., assembly language, the development of higher level language, FORTRAN (FORmula TRANslation) by IBM and suitable combinations of input, storage, processing and output units according to the requirement of a particular customer in second generation brought the computers close to ordinary man.

Examples of second generation computers are IBM 7000, NCR 304 (Nation Cash register),

IBM 650, ATLAS etc.

THIRD GENERATION (1965 - 70)

Further advances in computer technology took place around 1965 with the invention of circuit consisting of a number of transistors and other electronic components fused together on a single crystal. These circuits are produced either by Photographic Etching on crystals or by Metal-oxide-semi conductor (MOS) technology. In third generation computers, **Small Scale Integrated circuits** (SSI) and **Medium Scale Integrated circuits** (MSI) were used. The difference between SSI and MSI is only the number of units fused together to form a single unit. The third generation computers using integrated circuits proved to be highly reliable and relatively inexpensive. These computers were 10,000 times faster than first generation computers. Mass storage devices called Auxiliary or Secondary storage devices such as magnetic tapes, disks etc.,came into use. Thus there has been a vast improvement in all the units-input, storage & output, processing. Batch processing and time-sharing came into picture during third generation. In **Batch processing**, the data to be processed is gathered together in a group and is processed together as a single job. The next job, with its batched data, is processed when the previous job is completed. At peak processing times, there may be many jobs waiting to be processed.

Dr. John Kemeney, a mathematics professor at Dartmouth College, developed **Time sharing** software for implementation on General Electric 265 computer. With this concept, many terminals were attached to the central computer. The time-sharing software alternately served each user on a terminal.

Because of the speed at which the computer operated, each user felt that they had complete control on the computer, though many of them were using the computer at the same time. With the development of time-sharing, compiler processing became "Interactive".

Another major development in software during this period was **Multi-programming,** in which more than one program can be present in computer's memory simultaneously.

Many high level languages like BASIC (Beginner's All-purpose Symbolic Instruction Code), COBOL (Common Business Oriented Language) etc., were developed during this period. Also a new computer industry, called **software industry** was born during third generation to develop reliable software for different types of users.

IBM 360, ICL 2900, PDP-8, IBM-370 etc., were some of the third generation computers.

FOURTH GENERATION (1970-1985)

This period was a period of active research and development in many areas of computer technology. These computers used **Large Scale Integration (LSI) and Very Large Scale Integration (VLSI)** techniques. With VLSI technology, **micro-processors** (µP) came into existence. A micro-processor chip (approx 1/4" x 2" in size) contains an entire Central Processing Unit (CPU) on a single silicon chip. By the end of the decade a computer was available with processing speed ten times faster, with a memory capacity four times greater and with a size of 1/10th of the standard earlier IBM systems. The prices also reduced considerably. During 1970's, a major change took place relative to the way data was made available to a computer for processing. With the developments in both hardware and software, the seventies saw increased use of another mode of processing data : **transaction-oriented mode.** In this mode, data is entered into the computer system at the time of transaction rather than batching the transaction. Another major development was **data communications,** which allows computers to electronically communicate with each other and also allows a user to access a computer through a terminal located at a remote site. The data communication takes place over telephone lines or other communication methods.

IBM 4341, DEC 10, STAR 1000, PUP 11, APPLE II belong to the fourth generation of computers.

FIFTH GENERATION

A lot of research is going on fifth generation computers. The major shortcoming in the earlier generations was lack of thinking power. The fifth generation computers combine hardware and software to produce human intelligence. The institute of New Generation Computer Technology in Japan is planning to develop Knowledge Information Processing Systems based on innovative inference functions and technologies that meet the needs anticipated in the 1990's, including intelligent interaction between machine and inference using knowledge bases. Automatic programming, computational logic, pattern recognition will be extensively used in near future with the development of **Artificial Intelligence.**

EXERCISES

ESSAY TYPE QUESTIONS

1. Trace the historical evolution of the computer right from its initial stages to the development of the modern computers.
2. Why is Charles Babbage called the Father of modern computer science ? Name the machines associated with his name.
3. What do you understand by the term 'Generation' in computers ? Classify computers on this basis.

SHORT ANSWER QUESTIONS

4. Write a short note on ENIAC Computer.

5. Discuss briefly how Abacus was used for adding numbers.
6. Write a brief note on Leibnitz's Calculator.
7. Describe in a few lines, the Fourth Generation Computers.

OBJECTIVE TYPE QUESTIONS

8. Fill in the blanks with the correct alternative out of those given :

   (i) The first instrument invented by Chinese around 350 B.C. for calculations was.........(Pascal's machine, Abacus)
   (ii) ....was made up of gears which was used for adding numbers quickly. (Pascal's machine, Leibnitz's calculator)
   (iii) ....is known as Father of Computers. (Hollerith, Charles Babbage)
   (iv) ....made a machine to run on electricity for the first time. (Pascal, Hollerith)
   (v) First commercial computer developed in 1951 was..........(UNIVAC, ENIAC, EDSAC)
   (vi) The first generation computers used..........the second generation computers used...........and the third generation computers used.......chips. (transistors, vacuum tubes, silicon)
   (vii) Fourth generation computers used...........(vacuum tubes, large scale integrated circuits)
   (viii) In first generation computers, data and instructions were given in................by the programmer. (high level language, machine language)

9. Write 'True' or ' False' against each of the following.

   (i) ABACUS was invented by Japanese in 1200 A.D...............
   (ii) John Napier invented logarithms..........
   (iii) Herman Hollerith is known as Father of Computers...........
   (iv) A code of punches and no punches was invented by Herman Hollerith...........
   (v) The first generation computers used transistors.........
   (vi) The first commercial computer built by Sperry Rand Corporation was UNIVAC - I. ....
   (vii) EDVAC computer used the 'stored program' concept given by Von Neumann ....

10. Choose the correct answer from the following options :

    (i) A calculating device which consists of beads movable on rods is called
    a) Pascal's machine
    b) Abacus
    c) Leibnitz's calculator
    d) Babbage's analytical engine

    (ii) Vacuum tubes were used in
    a) Second generation computers
    b) First generation computers
    c) Third generation computers
    d) Fourth generation computers

    (iii) The first commercial computer made in the year 1951 was
    a) ENIAC
    b) UNIVAC
    c) EDVAC
    d) EDSAC

(iv) More than one program can be present in computer's memory in
   a) Time sharing
   b) Multiprogramming
   c) Batch processing

11. Match the following :

a) Storage	1) Father of computers
b) Abacus	2) Fortran
c) Logs and bones	3) Transistors
d) Babbage	4) Memory
e) Second generation computer	5) First calculating device
f) High level language	6) John Napier

12. Give full form of the following abbreviations :

(i) ENIAC
(ii) EDVAC
(iii) EDSAC

13. Answer in one word (or sentence) the following questions :

(i) Who invented logarithms ?
(ii) Which instrument is made of beads divided into two parts ?
(iii) Who invented the adding machine made up of gears ?
(iv) Who is known as the Father of Computers? What was his machine used for?
(v) Who invented a code of punches and no-punches ? What is his code called ?

2

BINARY STATE

In our day-to-day life we count numbers using the Arabic number system (decimal system). The decimal system utilizes ten symbols to represent the values. These symbols are 0,1,2,3,4,5,6,7,8 and 9. As there are ten symbols this system of representing numbers is called Decimal System. With the help of these ten symbols, any quantity can logically be represented. To represent quantities greater than 9, two or more symbols are used. Proper utilization of these symbols is based upon the concept of place values. In this system, each place position from right to left is as follows: ones, tens, hundreds, thousands, etc. These place values are obtained by using a base of 10 (because there are ten symbols) and raising the base to the next highest power each place position to the left starting with zero. The example, given below explains how decimal numbers derive their meaning.

The number seven hundred and seventy-five is represented by the digits 775. It derives its value from the place value of each digit. Thus the number 775 in the decimal system really means 7-seven hundreds, 7-seven tens and 5-five ones. The place value concept can be applied to a number system with any base.

To distinguish one number system from the other, the base value of the number will be written as a subscript. For example, 27_{10} means the number 27 is a decimal number.

THE BINARY NUMBER SYSTEM

The Binary number system utilizes only two symbols, 0 and 1. Thus, the Binary number system uses a base of 2 to obtain the place value of each digit. In Binary number system, each place position from right to left is as follows :

0	7	7	5	
THOUSAND	HUNDRED	TEN	ONE	PLACE VALUE
10^3	10^2	10^1	10^0	BASE

Explanation of how decimal numbers derive their meaning.

	0	0	0	0	1	
1 =	SIXTEEN	EIGHT	FOUR	TWO	ONE	PLACE VALUE
	2^4	2^3	2^2	2^1	2^0	BASE

Binary Representation of decimal number 1.

	0	0	0	1	0	
2 =	SIXTEEN	EIGHT	FOUR	TWO	ONE	PLACE VALUE
	2^4	2^3	2^2	2^1	2^0	BASE

Binary Representation of decimal number 2.

One, two, four, eight, etc. The place values are obtained by raising the base, i.e., 2, to the next higher power each place position to the left starting with zero. The above figure illustrates the decimal numbers, 1 and 2 represented in the Binary number system.

The following table shows the decimal numbers upto 20 and the binary numbers corresponding to them.

DECIMAL NUMBER	BINARY NUMBER
1	00001
2	00010
3	00011
4	00100
5	00101
6	00110
7	00111
8	01000
9	01001
10	01010
11	01011
12	01100
13	01101
14	01110
15	01111
16	10000
17	10001
18	10010
19	10011
20	10100

We, thus, have two number systems, one Decimal (with base 10) and the other Binary (with base 2). One can change the numbers written in one system to the other system.

CONVERSION OF NUMBERS (INTEGERS) FROM DECIMAL SYSTEM TO BINARY SYSTEM

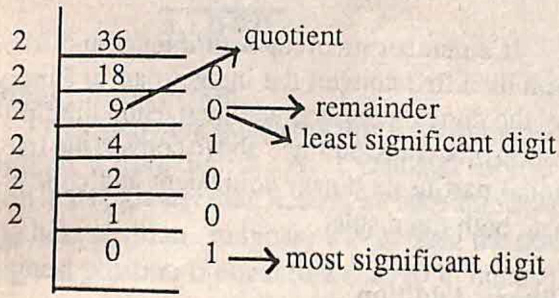
Working Method :

Divide the number in Decimal system by two and encircle the remainder. Divide the quotient by two and again encircle the remainder. Continue this process till quotient becomes zero.

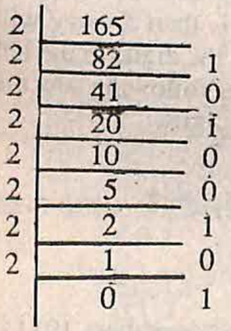
Now write the remainders left to right from bottom to top, that is the last remainder becomes the most significant digit and the first remainder becomes the least significant digit of the binary number.

Example :

Convert 36 and 165 from base 10 into base 2.



Hence $36_{10} = 100100_2$



Hence $165_{10} = 10100101_2$

CONVERSION OF NUMBERS FROM BINARY SYSTEM TO DECIMAL SYSTEM

To convert binary numbers to decimal numbers multiply each digit of the binary number with its place value and add all the products.

Example 1

Convert 100100 from the base 2 to the base10

Number	1	0	0	1	0	0
Place value	2^5	2^4	2^3	2^2	2^1	2^0

The number in decimal system will be obtained by multiplying the digits of the number with its place value and adding all the products, i.e.,

$$1 \times 2^5+0 \times 2^4+0 \times 2^3+1 \times 2^2+0 \times 2^1+0 \times 2^0$$
$$= 32+0+0+4+0+0 = 36$$

Hence $100100_2 = 36_{10}$

Example 2

Convert 1110001 from the base 2 to base 10.

Number	1	1	1	0	0	0	1
Place Value	2^6	2^5	2^4	2^3	2^2	2^1	2^0

∴ Its decimal equivalent is

$$1\times 2^6+1 \times 2^5+ 1 \times 2^4+0 \times 2^3+0 \times 2^2+0 \times 2^1+1 \times 2^0 =113$$

Hence $1110001_2 = 113_{10}$

Example 3

Convert 1111 from the base 2 to base 10.

Number	1	1	1	1
Place Value	2^3	2^2	2^1	2^0

Its decimal equivalent is

$$1 \times 2^3+1 \times 2^2+1 \times 2^1+1 \times 2^0 =15$$

Hence $1111_2 = 15_{10}$

REPRESENTATION OF FRACTIONS

The place values for decimal fractions from left to right are as follows: One tenth, one hundredth, one thousandth, etc. These can be represented as powers of 10 like :

10^{-1} , 10^{-2} , 10^{-3} , etc. For example 0.8125 means

$$8 \times 10^{-1} + 1 \times 10^{-2} + 2 \times 10^{-3} + 5 \times 10^{-4}$$

CONVERSION OF DECIMAL FRACTIONS TO BINARY FRACTIONS

The conversion of Decimal fractions to Binary fractions is based on the fact that a Decimal fraction is expressed as a sum of negative powers of 2. Successive multiplication of the Decimal fraction by 2 would give the coefficients of the negative powers of 2.

General Procedure

Multiply the fractional part by 2. The integer part can be zero or one. Encircle the integer part. Again multiply the new fractional part by two. Repeat the procedure till either the fractional part becomes zero or required number of digits after decimal part have been obtained. Writing from left to right, the integer part in the order of their creation is the binary equivalent. The decimal point is at the extreme left.

Example :

Convert 0.625 from base 10 to Binary fraction.

This can be obtained simply by multiplying 0.625 by 2 as shown below.

$$\begin{array}{lr} & .625 \\ & 2 \\ \hline (1) & .250 \\ & 2 \\ \hline (0) & .500 \\ & 2 \\ \hline (1) & .000 \end{array}$$

Hence the Binary equivalent of the Decimal fraction 0.625 is 0.101.

Example

Convert 0.2 from the base 10 to base 2.

$$\begin{array}{lr} & 0.2 \\ & 2 \\ \hline (0) & .4 \\ & 2 \\ \hline (0) & .8 \\ & 2 \\ \hline (1) & .6 \\ & 2 \\ \hline (1) & .2 \end{array}$$

It can be seen that the fractional part will never become zero. So we terminate the process of multiplication after obtaining the required number of digits.

Hence $0.2_{10} = 0.0011_2$

If a number involves both integer and fraction then first convert the integer part to Binary by the division method and then using multiplication procedure outlined above convert the fractional part to its binary equivalent and concatinate both the results.

Binary Addition

Binary addition is similar to the addition of Decimal numbers except that when the result of addition exceeds the value 1, then a carry will come into picture for adding the digits to the left of the current position. The following are the rules for adding two binary digits.

$0 + 0 = 0$

$0 + 1 = 1$

$1 + 0 = 1$

$1 + 1 = 10$

Example : Add the Binary numbers 101111 and 10111

$$\begin{array}{rl} 11111 & \text{carry} \\ 101111 & \\ 10111 & \\ \hline 1000110 & \end{array}$$

Explanation : First add the right most digits i.e.,1 and 1

$$1 + 1 = \overset{\text{carry}}{1}\underset{\text{result}}{0}$$

Now 1 will become the carry to the next addition and 0 is the result for this addition. Next we have to add 1,1 and 1. For this first add 1 and 1 and then add 1 to the result.

$$1 + 1 = 10$$
$$10 + 1 = 11$$

Now the result is one and carry is one. Continue this process until all the digits are added.

Example. Add the binary numbers 1110 and 1010

$$\begin{array}{rl} 1\,1 & \text{carry} \\ 1\,1\,1\,0 & \\ 1\,0\,1\,0 & \\ \hline 1\,1\,0\,0\,0 & \\ \hline \end{array}$$

Binary Subtraction

The following table explains the subtraction of digit Y from digit X. As in Decimal subtraction if Y is greater than X, then we take loan from the next position. In binary a loan from the next highest position is equal to 2 (i.e.,10 in binary)

X	Y	X-Y	
0	0	0	
0	1	1	after taking loan from the digit left to X.
1	0	1	
1	1	0	

Example : Subtract 011 from 101.

$$\begin{array}{rr} & 1\,0\,1 \\ - & 0\,1\,1 \\ \hline & 0\,1\,0 \\ \hline \end{array}$$

Subtraction of 1 from 1 yields 0 in the least significant position. To subtract 1 from 0 in the next position, we take a loan from 1 in next position which now becomes 10(=2) and the result is 1. The digit 1 after giving loan becomes 0. So finally subtracting 0 from 0 yields 0.

Example : Subtract 101100 from 110010

$$\begin{array}{rl} (1)\,(10) & \text{loan} \\ 1\,1\,0\,0\,1\,0 & \\ 1\,0\,1\,1\,0\,0 & \\ \hline 0\,0\,0\,1\,1\,0 & \\ \hline \end{array}$$

Binary Multiplication

Binary numbers are multiplied in the same manner as decimal numbers. The following rules are followed to multiply two binary numbers :

$$0 \times 0 = 0$$
$$0 \times 1 = 0$$
$$1 \times 0 = 0$$
$$1 \times 1 = 1$$

Example : Multiply 1101 and 111

$$\begin{array}{r} 1\,1\,0\,1 \\ 1\,1\,1 \\ \hline 1\,1\,0\,1 \\ 1\,1\,0\,1 \\ 1\,1\,0\,1 \\ \hline 1\,0\,1\,1\,0\,1\,1 \\ \hline \end{array}$$

Answer is 1011011

Example : Multiply 1010 by 101

$$\begin{array}{r} 1\,0\,1\,0 \\ 1\,0\,1 \\ \hline 1\,0\,1\,0 \\ 0\,0\,0\,0 \\ 1\,0\,1\,0 \\ \hline 1\,1\,0\,0\,1\,0 \\ \hline \end{array}$$

Answer is 110010.

Binary Division

Binary numbers are also divided in the manner similar to Decimal numbers. In Binary division the following rules are followed.

$$0/1 = 0$$
$$1/1 = 1$$

1/0 is not defined.

Example : Divide 11010 by 101.

$$\begin{array}{r|l} & 1\,0\,1 \\ \hline 101 & 1\,1\,0\,1\,0 \\ & 1\,0\,1 \\ \hline & 1\,1\,0 \\ & 1\,0\,1 \\ \hline & 0\,0\,1 \end{array}$$

The quotient is 101 and the remainder is 1

Example : Divide 110110 by 110

$$\begin{array}{r} 1001 \\ 110\overline{)\,110110} \\ \underline{110} \\ 110 \\ \underline{110} \\ 000 \end{array}$$

The quotient is 1001 and the remainder is 0.

OCTAL SYSTEM

The binary representation of numbers corresponding to the large numbers in the decimal system are very long strings of zeros and ones. To overcome this disadvantage a number system called Octal system is used to represent numbers. The Octal system uses digits 0, 1, 2, 3, 4, 5, 6, 7 and thus utilizes a base of 8 to obtain the place value of each digit. The place values in an Octal system are as follows : one, eight, sixty-four, etc.

Conversion of numbers (Integers) from Decimals to Octal.

The conversion is done exactly in the same way as it is done in binary except that the divider in octal system is 8.

Example : Convert 372 from base 10 into base 8.

$$\begin{array}{r|l c c} & & & \text{remainders} \\ 8 & 372 & & \\ \hline 8 & 46 & - & 4 \\ \hline 8 & 5 & - & 6 \\ \hline & 0 & - & 5 \end{array}$$

Hence $372_{10} = 564_8$

HEXADECIMAL SYSTEM :

The Hexadecimal system uses sixteen digits starting with zero. The representation of the digits 0 to 9 are same as decimal digits. To represent the decimal values 10-15, hexadecimal system uses the letters of alphabet A to F. Thus the decimal value 10 is represented in Hexadecimal as A. The following table illustrates the Hexadecimal equivalents of decimal numbers 0 to 15.

Decimal Value	Hexadecimal Representation
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	A
11	B
12	C
13	D
14	E
15	F

Conversion of numbers (Integers) from Decimal to Hexadecimal

The conversion is done exactly in the same way as it is done in Octal except that the divider here is 16.

Ex : (1) Convert 16 from base 10 to base 16.

$$\begin{array}{r|c c} 16 & 16 & \\ \hline 16 & 1 & 0 \\ \hline & 0 & 1 \end{array} \qquad \text{Ans}=(10)_{16}$$

(2) Convert 46 from base 10 to base 16.

$$\begin{array}{r|c c} 16 & 46 & \\ \hline 16 & 2 & E \\ \hline & 0 & 2 \end{array} \qquad \text{Ans}=(2E)_{16}$$

(3) Convert 460 from base 10 to base 16.

$$\begin{array}{r|c c} 16 & 460 & \\ \hline 16 & 24 & C \\ \hline 16 & 1 & 8 \\ \hline & 0 & 1 \end{array} \qquad \text{Ans}= (18C)_{16}$$

Conversion of numbers from Octal, Hexadecimal to decimal.

To convert numbers from Octal and Hexadecimal to Decimal exactly the same procedure is followed as in Binary.

Ex: (1) Convert 367 from the base 8 to the base 10.

Number 3 6 7
Place Value 64 8 1

The decimal equivalent is

$$3 \times 64 + 6 \times 8 + 7 \times 1 = 237$$

Hence $367_8 = 237_{10}$

(2) Convert 256 from the base 8 to the base 10.

Number 2 5 6
Place Value 64 8 1

The decimal equivalent is

$$2 \times 64 + 5 \times 8 + 6 \times 1 = 174$$

Hence $256_8 = 174_{10}$

Ex: (1) Convert 4CD from the base 16 to the base 10

Number 4 C D
Place value 256 16 1

The decimal equivalent is

$$4 \times 256 + C \times 16 + D \times 1 \text{ or } 4 \times 256 + 12 \times 16 + 13 \times 1 = 1229$$

Hence $4CD_{16} = 1229_{10}$

(2) Convert A35B from the base 16 to the base 10

Number A 3 5 B
Place Value 4096 256 16 1

The decimal equivalent is

$$A \times 4096 + 3 \times 256 + 5 \times 16 + B \times 1 \text{ or } 10 \times 4096 + 3 \times 256 + 5 \times 16 + 11 \times 1 = 41819$$

Hence $A35B_{16} = 41819_{10}$

Conversion of Numbers (Integers) from Binary to Octal, Hexadecimal and vice-versa.

We know that the base of Binary number system is 2, Octal number system is 8 ($2 \times 2 \times 2$) or three 2's and Hexadecimal system is 16($2 \times 2 \times 2 \times 2$) orfour 2's. The Binary numbers can be converted into octal or hexadecimal by grouping successively 3 or 4 digits of the Binary number starting from the right and renaming the group by the symbol in the Octal or Hexadecimal system. Similarly if the number is in Octal or Hexadecimal system then it can be expanded in terms of 3 or 4 binary digits to represent it in Binary. The following table shows the equivalence between Binary, Hexadecimal and Octal systems.

Binary numbers and their Octal equivalents.

Binary	Octal	Binary	Octal
000	0	101	5
001	1	110	6
010	2	111	7
011	3		
100	4		

Binary numbers and their Hexadecimal equivalents.

Binary	Hexa-decimal	Binary	Hexa-decimal
0000	0	1000	8
0001	1	1001	9
0010	2	1010	A
0011	3	1011	B
0100	4	1100	C
0101	5	1101	D
0110	6	1110	E
0111	7	1111	F

Converting Binary numbers to Octal systems.

Make the digits in Binary number as groups of 3 digits each starting from right. If the last group falls short of 3 digits, fill it with zeros and then convert into decimal number treating each group as an independent binary number and write the digits in the order in which the groups are. The resulting number is the Octal equivalent of the Binary number.

Ex: Convert 1110010010 from base 2 to base 8.

The above number can be grouped as :

$$\underline{1} \quad \underline{110} \quad \underline{010} \quad \underline{010}$$

This will become :

$$\frac{001}{1} \quad \frac{110}{6} \quad \frac{010}{2} \quad \frac{010}{2}$$

Hence the Octal equivalent of $(1110010010)_2$ is 1622.

S.C.E.R.T. W.B. LIBRARY
Date 5.11.98
9758

S.C.E.R.T. W.B. Library

Converting Binary numbers to Hexadecimal system.

This is same as converting to base 8 except that each group here consists of 4 digits and the decimal numbers 10 to 15 obtained during conversion are represented by the letters A to F.

Ex : Convert 1110010010 from base 2 to base 16

The above number can be grouped as :

$$\begin{array}{ccc} 11 & 1001 & 0010 \end{array}$$

This will become :

$$\begin{array}{ccc} \underline{0011} & \underline{1001} & \underline{0010} \\ 3 & 9 & 2 \end{array}$$

Hence the hexadecimal equivalent of (111 0010010) is 392.

Converting Octal numbers to Binary numbers

To convert an octal number to its binary equivalent, convert each digit of the Octal number into Binary equivalent with three digits. If the number of digits in Binary are less than 3 for any Octal digits then add zeros on the left side to make them 3, and concatenate these primary strings in the order in which they are in Octal number.

Ex: Convert 370 from base 8 to base 2

$$\begin{array}{cccl} \underline{3} & \underline{7} & \underline{0} & \\ 011 & 110 & 000 & \longrightarrow 011110000 \end{array}$$

Hence the Binary equivalent of the Octal number 370 is 011110000.

Converting Hexadecimal numbers to Binary numbers.

Conversion from Hexadecimal to Binary is same as Octal except that each digit in hexadecimal is converted into 4 digit binary number.

Ex: Convert ABC from Hexadecimal to Binary.

$$\begin{array}{cccl} \underline{A} & \underline{B} & \underline{C} & \\ 1010 & 1011 & 1100 & \rightarrow 101010111100 \end{array}$$

Hence the binary equivalent of the Hexadecimal number ABC is 101010111100.

Converting Decimal fractions to Octal and Hexadecimal.

The conversion procedure from Decimal to Octal and Hexadecimal is same as for Binary except that instead of multiplying by two we multiply by 8 and 16 respectively.

Ex : Convert 0.815 from base 10 to base 8.

$$\begin{array}{lr} & 0.815 \\ & \underline{\quad\quad 8} \\ (6) & .520 \\ & \underline{\quad\quad 8} \\ (4) & .160 \\ & \underline{\quad\quad 8} \\ (1) & .280 \\ & \underline{\quad\quad 8} \\ (2) & .240 \\ & \underline{\quad\quad 8} \\ (1) & .920 \\ & \underline{\quad\quad 8} \\ (7) & .360 \text{ and so on} \end{array}$$

Hence the Octal equivalent of 0.815 in Decimal is 0.641217 (up to 6 places)

Ex: Convert 0.225 from base 10 to base 16.

$$\begin{array}{lr} & 0.225 \\ & \underline{\quad\quad 16} \\ (3) & .600 \\ & \underline{\quad\quad 16} \\ (9) & .600 \\ & \underline{\quad\quad 16} \\ (9) & .600 \end{array}$$

Hence the Hexadecimal equivalent of 0.225 in Decimal is 0.3999...

EXERCISES

ESSAY TYPE QUESTIONS

1. Describe the Decimal System of numbers with examples.
2. Describe the Binary Number System with examples.
3. How does a Hexadecimal System differ from Octal System of numbers?
4. Explain with examples conversion of Hexadecimal number into Binary number.

SHORT ANSWER QUESTIONS

5. Perform the following additions :
   (*i*) $(1011001)_2 + (1011)_2$
   (*ii*) $(111)_2 + (101101)_2$
   (*iii*) $(1011010)_2 + (10000)_2 + (1011)_2$
   (*iv*) $(111111)_2 + (110000)_2 + (11110)_2$
   (*v*) $(101110)_2 + (11011)_2 + (11110)_2$

6. Perform the following multiplications :
   (*i*) $(11101)_2 * (10101)_2$
   (*ii*) $(1100)_2 * (1010)_2$
   (*iii*) $(110111)_2 * (11111)_2$
   (*iv*) $(11111)_2 * (1011)_2$
   (*v*) $(10101)_2 * (10101)_2$

7. Perform the following divisions :
   (*i*) $(1100)_2/(10)_2$
   (*ii*) $(101100)_2/(100)_2$
   (*iii*) $(1110011)_2/(1011)_2$
   (*iv*) $(1001110)_2/(110)_2$
   (*v*) $(110010)_2/(101)_2$

8. Perform the following subtractions :
   (*i*) $(11110)_2 - (100)_2$
   (*ii*) $(101011)_2 - (1100)_2$
   (*iii*) $(100110)_2 - (10010)_2$

9. Perform the following conversions :
   (*i*) $(1100101)_2 = (\quad)_{10}$
   (*ii*) $(1111010)_2 = (\quad)_{10}$
   (*iii*) $(968)_{10} = (\quad)_2$
   (*iv*) $(453)_{10} = (\quad)_2$
   (*v*) $(101010111)_2 = (\quad)_{16}$
   (*vi*) $(1101101)_2 = (\quad)_8$
   (*vii*) $(98)_{10} = (\quad)_8$
   (*viii*) $(126)_{10} = (\quad)_{16}$

(ix) $(2BC)_{16} = (\quad)_{10}$

(x) $(67)_8 = (\quad)_{10}$

10. Perform the following additions:

(i) $(1010.01)_2 + (10.11)_2$

(ii) $(1111.0011)_2 + (0.1101)_2$

(iii) $(10000.1101)_2 + (1010111.000111)_2$

(iv) $(101.101)_2 + (110.1)_2$

(v) $(110.011)_2 + (111.101)_2 + (100.01)_2$

OBJECTIVE TYPE QUESTIONS

11. Fill in the blanks:

(i) The Decimal system uses .... symbols to represent the values.

(ii) The Binary number system utilizes only two symbols .... and .... to represent the values.

(iii) If $(1100)_2 = (12)_{10}$ then $(111001)_2 = (\quad)_{10}$

(iv) $(1)_2 + (1)_2 + (1)_2 = (\quad)_2$

(v) $(10)_2 * (\quad) = (100)_2$

12. Write 'True' or 'False' against each of the following:

(i) $(110)_2 = (8)_{10}$. __________

(ii) An Octal Number System consists of eight digits to represent the values. __________

(iii) (7A2) is a number in Octal number system. __________

(iv) In Hexadecimal System there are 16 symbols to represent the values. __________

(v) Conversion of Binary numbers to Decimal numbers is done by grouping the Binary digits. __________

☐☐

3

COMPUTERS

The most obvious question concerning computers is, "What is a computer?" A computer is an electronic machine, operating under the control of instructions stored inside its memory. As an aid to problem solving, it accepts data, both numeric and non-numeric, processes it, and presents it in a desired form.

TYPES OF COMPUTERS

Analog Computers

Though generally not regarded as such, the most prevalent computer is the simple mechanical analog computer, in which gears, liners, ratchets, and pawls perform mathematical operations. The automobile speedometer, the water and electricity meters are some of the simple examples of Analog devices. Another example of an Analog device is clock with a dial and two hands.

An Analog computer operates by measuring voltages and currents. It works on the supply of continuous electrical signals. The display of output is also continuous that is, it uses inputs that are proportional to the instantaneous value of variable, combines them in a pre-determined way, and produces outputsthat are a continuously varying function of the inputs and the processing.

This machine has a very limited memory. Also it can perform a limited type of calculations. Although Analog computer performs calculations at a high speed the accuracy of this machine is very low.

Digital Computers

As the name suggests they operate on quantities represented as digits. They operate on discrete quantities. In digital computer, both numeric and non-numeric information is represented as strings of digits. These computers use binary numbers 0 and 1 to represent the information. The information is given to the computer in the form of discrete electrical signals.

Some examples of mechanical digital computer are desk calculators, office adding machines etc. Electronic digital computers include a large number of calculators and the present day computers.

Hybrid Computers

Computers which possess the features of both Analog computers as well as Digital computers are called Hybrid computers. They allow the user to process both continuous and discrete information.

Computers can also be classified according to their storage capacities and speed. Accordingly, they can be categorised as Super Computers, Main Frame Computers, Mini Computers and Micro Computers. A brief description of these is given below:

Super Computers

The storage capacity of these computers is in the range of 64 MB to 4 GBand the speed of the CPU is 100 mips. The word length is from 64 to 128 bits. High speed laser printers, disks, tapes etc., are used as peripherals. These are the fastest computers made to solve a range of large scale problems which require large numeric computations, such as weather prediction, complex molecular structural calculations, etc.

Typical examples of this type are CRAY-2, CYBER 205 etc.

Main Frame Computers

These computers are supported by large memory ranging from 16 MB to 128 MB and the speed of the CPU is 1 TO 10 mips. The word length is in the range of 32 to 64 bits. These machines use chain printers, 1000 MB disks, high speed tapes and 32 to 128 user terminals as peripherals. These are used to solve all types of problems. A few important applications in which these computers are used are large engineering designs, line applications with large data bases etc. DEC 1090, Cyber 170, IBM 360/370, UNIVAC 1100/60 etc., are a few examples of main frame computers.

Mini Computers

The storage capacity of these computers is in the range of 2 MB to 16 MB. The speed of the CPU is five times that of a micro computer (500 kips). The word length ranges from 16 to 32 bits. The main advantage of these computers is more than one user can use it at the same time by attaching a number of terminals. In addition to the peripherals used by the micro computers, these can also use line printers, magnetic tapes, exchangeable hard disks etc. These are used for a variety of applications; process control, Computer Aided Design, Scientific research, On line reservation systems etc., to name a few. These computers accept all high level languages to process data.

PDP-II, VAXII/780, TDC-332 etc., are a few examples of mini computers.

Micro Computers

These computers now a days are well known by the name PC. Micro computers contain all parts of CPU on a single Chip. The storage capacity of these computers is low and upto a maximum of 2MB. The speed of the CPU is only 100 kips (kilo instructions per second). In general, the maximum word length of these computers is 16 bits. But most of the computers use 8-bits words. The number of peripherals attached to these computers is very few, for example floppy disk drive, character printer, VDU etc. They are mostly used in departmental stores, small companies for payroll etc., and by professional users like engineers, lawyers and accountants. It is for this reason these computers are also known as personal computers (PC).

A few examples of micro computers are Commodore, Uptron S-800, DCM Tandy, IBM-PC etc. The low costs and the capability to work with different high level languages have made these machines more popular.

EXERCISES

ESSAY TYPE QUESTIONS

1. What are the three basic types of computers? Explain them stating clearly the difference between them.
2. Into how many categories can you classify the computers according to their size? Explain giving examples.
3. Differentiate between Super Computers and Micro Computers giving examples of each.

SHORT ANSWER QUESTIONS

4. Give some important differences between a digital computer and an analog computer.
5. Write a short note on Mini Computers.
6. Describe in brief the Main Frame Computers giving examples.

OBJECTIVE TYPE QUESTIONS

7. Fill in the blanks :
   - (i) A......................is an electronic machine, operating under the control of instructions stored inside its memory.
   - (ii) A computer in which gears, lines, ratchets, and pawls perform mathematical operations is called...........................
   - (iii) An Analog Computer operates by measuring voltages and ....
   - (iv) In ....computers the information is given in the form of discrete electrical signals.
   - (v) A computer which possesses the features of both Analog Computers and Digital Computers is called ....

8. Answer the following questions in a word or a sentence :
   - (i) Give two examples of Analog Computers.
   - (ii) Which computer operates on discrete quantities?
   - (iii) Give two examples of Digital computers.
   - (iv) Give the names of four categories into which Computers can be divided according to their size.
   - (v) Which class of computers is now a days known as PC?
   - (vi) What is the maximum storage capacity of a micro computer?
   - (vii) What is the maximum wordlength of micro computers?
   - (viii) Give any two examples of micro computers.
   - (ix) What is the storage capacity of Mini Computers?
   - (x) What is the word length of Mini Computers?
   - (xi) Give two fields in which Mini Computers are used.
   - (xii) Give three examples of Mini Computers.
   - (xiii) What is the memory size of Main Frame Computers?
   - (xiv) What is the speed of the CPU of these computers?
   - (xv) What type of printers are used by Main Frame Computers?
   - (xvi) Give three examples of Main Frame Computers.
   - (xvii) What is the storage capacity of Super Computers?
   - (xviii) What is the speed of the CPU of these Computers?
   - (xix) Give two applications of Super Computers.
   - (xx) Give two examples of Super Computers

4

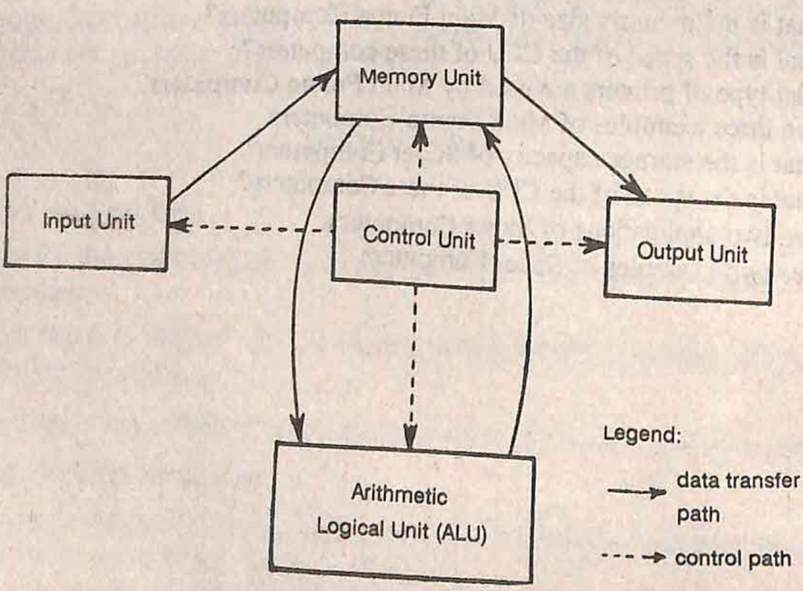
STRUCTURE OF THE COMPUTER

The hardware of a computer system refers to the physical components made of electronic, electrical and mechanical devices. On the other hand, software of a computer system refers to programs written for the effective utilisation of the hardware. Hardware and software together form a **Computer System.**

A computer is typically equipped with four important parts, namely, Input Unit, Central Processing Unit, Output Unit and Memory Unit. The Central Processing Unit (CPU) consists of two parts, namely, **Arithmetic and Logic Unit** (ALU) and **Control Unit** (CU). In addition to the above five basic units, computers also employ secondary memory units which are also called **Auxiliary Memory** or backing memory for holding information for longer periods. The input and output units are abbreviated as I/O devices and are called **Peripheral Devices.** Usually, all the units which are not part of the Central Processing Unit but are connected to CPU are called Peripherals. This means even the secondary memory units are also called peripherals.

INPUT UNITS

Input devices are used to transfer the data and instructions to the computer. These devices translate the information from a language in which the user is working, into a language which the computer can understand. Usually this language is in the form of binary codes. Most commonly used input devices are Key board,



Functional block diagram of a computer

Card readers, Magnetic Disc and Floppy Disc readers. The other devices which are used as input devices are Paper Tape Reader, Visual Display Unit (VDU), Magnetic Tape Reader, Mouse, Joystick, etc.

CENTRAL PROCESSING UNIT

As already described it consists of two parts, ALU and Control Unit.

Control Unit (CU)

Control Unit is the nerve centre of the computer. The control unit directs and coordinates the activities of the entire computer. Its tasks include controlling the input/output units, controlling the operations of the arithmetic & logic unit and transferring data to and from main memory. An important function of CU is program execution, i.e., fetching an instruction from memory, decoding it and sending it to appropriate place in the computer to get it executed. For example, CU ensures that both operands have reached ALU before any operation begins in ALU. CU gets program instructions from memory and executes them one after the other. After getting the instruction from the memory the instruction is decoded, i.e., it is decided which operation is to be performed and then the operation is carried out and the result will be sent back to memory if necessary. After completing the work of this instruction, CU sends signal to memory to send the next instruction to CU. It controls the input/output devices by controlling the flow of data among them, i.e., from input devices to memory and from memory to output devices.

Arithmetic and Logic Unit (ALU)

The arithmetic and logic unit performs **Arithmetic operations** such as addition (+), subtraction (-), multiplication (*), division (/), and exponentiation (↑ or * *). While performing an arithmetic operation the operands are brought to A.L.U. from the memory. In ALU the required operation is done and the result is sent back to the memory for storage. All this happens under the control of control unit.

ALU also performs comparisons which are called **Logical Operations.** The logical operations occur when the operands are connected with logical operators. Equal to (=), not equal to < >, less than (<) and greater than (>) are logical operators.

Suppose we have to compare two numbers for equality. First the two numbers are brought to the ALU and they are compared for equality. The outcome of the comparison is either True or False. If the two numbers are equal then the result is True, otherwise False. This result is used to control the flow of the program, i.e., to decide which instruction is to be performed next. The result can also be stored in memory for later use. These operations provide the capability of decision making through the computer.

MEMORY

Memory can be divided into Main Memory (or Primary Memory) and Secondary Memory.

Main Memory

Main Memory of a computer provides storage for data and instructions on which Control Unit and Arithmetic and Logic Unit operate. When data is entered into a computer from an input unit, it is stored in main memory. It holds the program and data currently being used for the computer. It also stores the intermediate results.

Memory is divided into a number of locations which are called Memory Cells. Each cell can store a fixed amount of information. Each memory location has a unique address. By specifying this address one can retrieve information from that location. If one wants to store information in a cell, which already contains some information, then the old information will be completely erased and the new one will be placed in that particular location. The information stored in main memory gets erased when the power is switched off. In other words this type of memory is volatile in nature.

The capacity of main memory is limited and the operations involved with main memory are very fast. Generally Semi Conductor memories or Magnetic Bubble memories are used as main memory and the cost of these memories is very high.

The main memory usually contains two parts, Random Access Memory (RAM) and Read Only Memory (ROM). The RAM is volatile in nature and all user programs and data are stored in this part of the main memory. On the other hand ROM is non-volatile in nature and all the instructions required by the computer when it is switched on are placed in this part of the main memory. It also contains the programs for calculating functions, such as SIN, COS, LOG, square root etc., which are used frequently in programming and the user need not write programs for them. From this part one can only read the information which is already there and it is not possible to write new information into it.

Secondary Memory

Because of the volatile nature and limited capacity of the Main Memory, the secondary memory units become part of the computer. Secondary memory devices are non-volatile in nature, i.e., the information stored on them cannot be erased even if we cut off the power supply. Therefore, after working on the computer, the program can be stored on the secondary device so that it can be used later. But for using this program again on the computer, this program from secondary storage device has to be copied on the main memory. When this is done, the program is in the main memory as well as in the secondary memory, so that even if during execution of the program if the power goes off, again the program can be copied from the secondary memory. Thus secondary memory are used to store the information but the information can not be used directly by the control unit. This information has to be transferred first to the main memory. These devices can store large amount of information. The rate at which the information is transferred to main memory and vice versa is called **Transfer rate.**

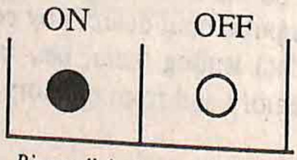
Various devices used for this purpose are magnetic tapes, magnetic drums, floppy disks, hard disk, cassette recorders etc.

OUTPUT UNITS

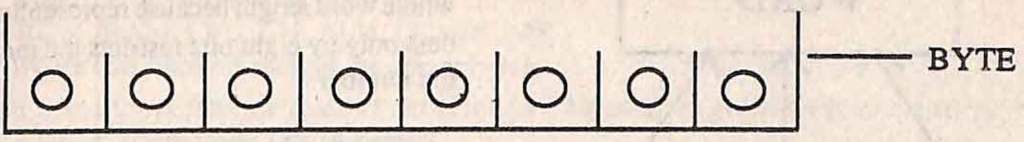
Output devices are used to get output or processed information from the computer. All the information inside the computer is in the form of binary digits (0 and 1). Output devices convert them to numbers and words which are easier to understand. Output devices can also generate graphs, sound and motion. Most commonly used output devices are printers and visual display unit (VDU), the television-like screen etc.

COMPUTER ORGANISATION

In the previous sections we saw that the memory of a computer was divided into cells each having unique address. These cells are further divided into smaller parts known as bits. The word bit is a contraction of two words BInary digiT. A bit can represent one of the two values - zero (0) and one (1). A bit which contains the value zero is said to be "off", and a bit which contains the value one is said to be "on". The computer will determine the status of "off" or "on" with the help of electronics in the hardware. Regardless of the electronic means used to detect whether a bit is "on" or "off", it is the ability of the electronic circuitry to detect the status of a bit which is the basis for storing data in computer memory.



Binary digit showing the value ON & OFF



1 BYTE

A single bit only cannot be used to represent the decimal digits zero to nine, alphabets A to Z or special characters like +,-,*,/, etc. For a computer memory to be useful it must be able to store numbers, alphabets and special characters. To do this a series of bits are used. Many computers use a combination of eight bits as a unit of storing data. These eight bits are called a **byte.**

Thus a byte is a location in computer memory consisting of eight adjacent bits. A combination of bits 'on' and bits 'off' is used to represent and store numbers, alphabets and special characters. Codes are assigned to the various combinations which can be obtained from the series of bits being 'on' or 'off'. Hence, one byte is the smallest unit which can represent a data item or character.

Extended Binary Coded Decimal Interchange Code (EBCDIC)

One of the most commonly used coding schemes, called the Extended Binary Coded Decimal Interchange Code, uses eight bits to store a letter of the alphabet, number or special character. These eight bits are together called a 'byte'. Thus, as a character is read into main computer storage, it is stored in a byte of storage. The electronics of the computer system interpret the character which is read from an input device and form the correct bit pattern to represent the character. This bit pattern is stored in a byte of storage. The electronics of the computer system interpret the character which is read from an input device and form the correct bit pattern to represent the character. This bit pattern is stored in a byte and when the byte is referenced to a computer instruction, the correct character will be processed.

With EBCDIC, each byte is divided into two portions—the zone portion and the digit portion. Numeric values are represented in EBCDIC *with all* the zone bits "on" and the proper combination of bits in the digit portion of the byte 'on' to represent the particular numeric value. For example, value zero is represented as shown below :

O =

1	1	1	1	0	0	0	0
●	●	●	●	○	○	○	○
8	4	2	1	8	4	2	1
Zone				Digit			

Alphabetic data is represented in EBCDIC with a combination of bits "on" and "off". For example letter A is represented as shown below :

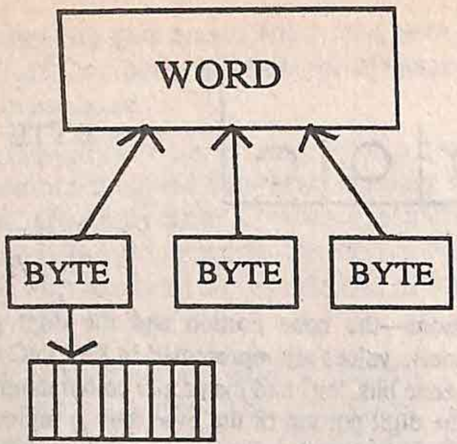
A =

1	1	0	0	0	0	0	1
●	●	○	○	○	○	○	●
8	4	2	1	8	4	2	1
Zone				Digit			

Special characters like ? @ etc. are also represented with a particular combination of bits "on" and "off"

A computer word, like a byte, is a group of fixed number of bytes. The number of bytes in a word varies from computer to computer but is fixed for each computer. The length of a computer word, usually called word length is in the range of 1 byte to 64 bytes. **Word length** implies the number of bits which can be handled in a simple operation. Thus higher the word length more information can be tackled simultaneously.

A computer stores information in the form of computer words. Each cell represents a word and many have a length of 16 bits, 32 bits, 48 bits, 64 bits and so on. The most common word length is 32 bits or 4 bytes. The three level construction of a word is as follows :



Level Construction of word

If a computer word is 24 bits long and if each data item needs 8 bits to represent it fully, then a computer word can accommodate three such data items. Generally, each non-numeric character needs 8 bits to represent it. But numeric data uses whole word length because representing numeric data only by eight bits restricts the magnitude of the number.

Computer memory does not contain just single word of memory but is composed of many words. The size of the memory is normally expressed in terms of the number of **KILO-BYTES.** The alphabet K is used to represent Kilobyte. One Kilobyte is equal to 1024(=2^{10}) bytes. For example, the size of memory for a personal computer could be expressed as 32K, meaning the computer contains thirty-two 1024 byte units of memory which is equal to 32768 bytes. The size of the computer's memory is also expressed in bigger units such as Megabytes (1 MB = 2^{20}bytes) and Gigabytes (1 GB = 2^{30}bytes)

EXERCISES

ESSAY TYPE QUESTIONS

1. Explain in about 200 words the basic structure of a computer along with its diagram.
2. Explain with examples the memory unit of a computer.
3. Explain in brief the process of reading information from the main memory.

SHORT ANSWER QUESTIONS

4. Differentiate between the following :
   - (i) Volatile and non-volatile memory
   - (ii) ROM and RAM
   - (iii) Hardware and Software
5. Write a short note on INPUT/OUTPUT Units.
6. What are the main functions of Arithmetic and Logic Unit (ALU) and Control Unit (CU) ?

OBJECTIVE TYPE QUESTIONS

7. Fill in the blanks :
   - (i) In general a byte consists of ....bits.
   - (ii) Semiconductor memory is a ....memory.
   - (iii) The information from auxiliary memory is transferred to ALU via..........................memory.
   - (iv) Instructions transferred from main memory are termed as ....
   - (v) The memory location address of main memory of size 64K unit of a computer varies from ....to...........................
   - (vi) RAM stands for..

8. Give the full form of the following :
ROM, RAM, CPU, CU, ALU, I/O, KB, MS
Write two lines on each of the above items.

9. Choose the correct alternatives out of the four alternatives given for each question. One or more alternatives can be correct.

(i) A computer system consists of
  (a) Input, Output unit
  (b) Memory Unit
  (c) Arithmetic and Logic unit
  (d) Central processing Unit

(ii) Main memory unit of a computer
  (a) Stores bulk of data and instructions
  (b) Stores small amount of data and information
  (c) Performs arithmetic calculations
  (d) Supervises the working of all units.

(iii) Auxiliary memory transfers information to
  (a) A.L.U.
  (b) Output unit
  (c) Control unit
  (d) Main memory unit

(iv) The main memory of a computer as compared to secondary memory is
  (a) Fast
  (b) Slow
  (c) Large
  (d) Cheap

(v) In a multiprocessing computer architecture
  (a) One program is handled at a given instant
  (b) Many programs are handled at a given time
  (c) Only two programs are handled at a given time
  (d) None of these.

10. Write 'True' or 'False' against each of the following :
(i) Input unit transfers a program to A.L.U.........................

(ii) In a computer RAM is used as short memory because it is volatile................

(iii) In a computer, user programs are stored in ROM.................................

(iv) A bit that contains the value zero is said to be "ON"..................................

(v) A single bit only cannot be used to represent more than one character................

11. Write in two or three sentences the functions of following units of a computer system:
(1) Input (2) Output (3) Arithmetic and Logic (4) Memory (5) Control

12. How are bit, byte and word related to each other?

5

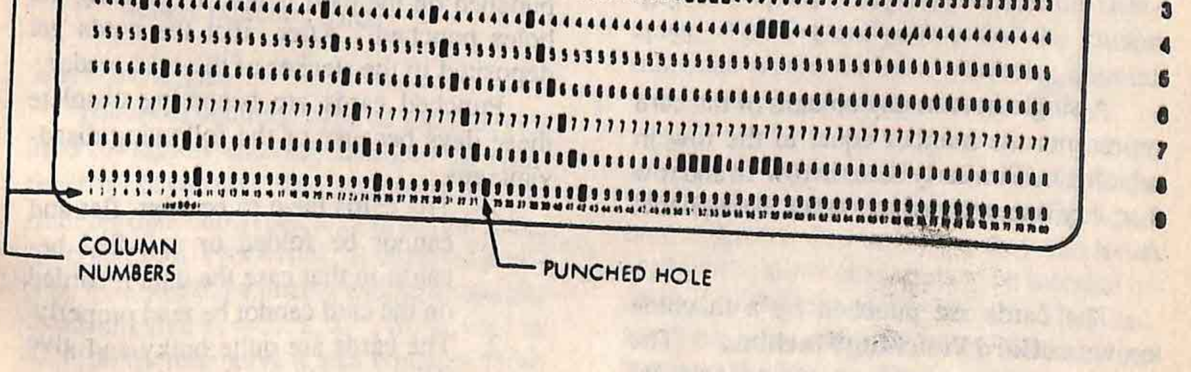
INPUT/OUTPUT DEVICES

Basically the operations that a computer system performs are input, processing and output. CPU accepts program and data from its input units and produces results on output units. These input and output units are commonly abbreviated as I/O units and are also called **Peripherals.** Most commonly used input media have been Punched cards, Punched Paper tape, Magnetic tape, Floppy disc etc. The data recorded on these is transmitted to the computer using input devices such as Card reader, Paper tape reader, Tape drives and Disk drives. Once the data is inside the computer it will be processed and the results are obtained/printed in the desired form on the output devices. Dot Matrix Printer, Line Printer, Visual Display Unit, Teletypewriter, Plotter etc., are a few commonly used output devices. Now let us study some of the important data recording media and I/O devices.

INPUT DEVICES

(1) PUNCHED CARDS

The standard punched card has 80 columns and its dimensions are 19 cm x 8 cm x 0.012.



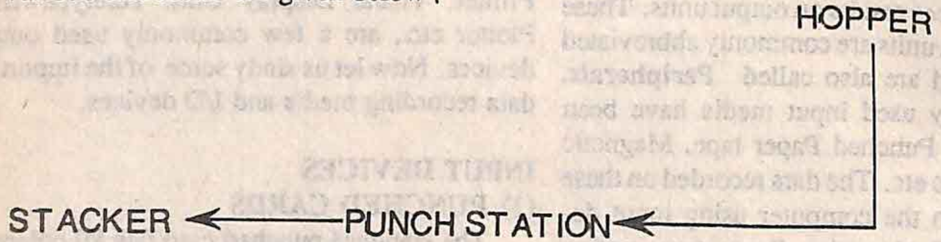
Schematic of a punched card

The top left most portion has a corner cut. The purpose of the cutting is to make sure that all cards are facing the same way. The card has 12 rows or punching positions. The rows 11 and 12 are on the top of the card and are called **zone punches.** The remaining ten rows are numbered as 0 to 9 from top to bottom below the rows 11 and 12. The rows 1 to 9 are called **digit punches.** The row 0 is used both for zone punch and digit punch. The codes developed by Dr. Hollerith for various characters are given below :

Number	Code	Letter	Code
0	0	A	12-1
1	1	B	12-2
2	2	C	12-3
3	3	D	12-4
4	4	E	12-5
5	5	F	12-6
6	6	G	12-7
7	7	H	12-8
8	8	I	12-0
9	9	J	11-1

A single hole in any column of the card represents the number equal to the row in which it is. Similarly holes in row 12 and row 1 in any one column represent the alphabet A.

The cards are punched by a machine known as **Card Punching Machine.** The key board of the punching machine is similar to that of an ordinary typewriter. The major difference lies in the location of the special characters. There is a hopper to hold the cards to be punched. When the user depresses the **FEED** key, the hopper releases a card to the punch station. When the **REGISTER** key is pressed the card is ready to be punched. If punching of a card consists of characters less than 80 then by pressing **"REL"** (Release) key, the card moves out of the punch station and will be stacked in a stacker.



Outline representation of Card Punching Machine

The punched cards together with the card (usually given from the computer centre and is placed at the end to distinguish between one bunch of cards and another) are then batched together and fed to the computer through an input device called **card reader.** A card reader consists of a card hopper in which the cards are loaded. Then card reader moves one card at a time in the serial order past a read station where the information punched on the card is read by sensing the holes punched. After that the cards get deposited in the stacker of the card reader.

Punched cards are becoming obsolete these days because of the following disadvantages.

1. The cards have to be kept flat and cannot be folded or spindled because in that case the data recorded on the card cannot be read properly.
2. The cards are quite bulky and give rise to problems of proper storage.

3. Cards are made of card board which absorbs moisture and expands in hot weather.
4. The system on the whole is very slow, bulky and easily perishable.
5. Incorrectly punched cards cannot be corrected and have to be replaced by newly punched cards.

(2) KEY BOARD.

The most common type of input device used for rapidly entering data into the computer is keyboard. The keyboard used to enter the data into a computer is similar to the keyboard of a typewriter. It contains numbers, letters of the alphabet and some special characters like +, —, /, £, $, ", &, %, etc. In addition to these keys it also contains some special purpose keys.

Two of the most widely used special purpose keys are **cursor control** keys. These are usually marked with arrows for four directions, i.e, left, right, up and down. These keys can move the cursor around the screen of the terminal. A cursor is a symbol, such as a square character or underline character. The present position of the cursor indicates where the next character entered will be displayed on the screen.

Key Board

Many keyboards have special keys which can be used to alter the text displayed on the screen. For example, they can have keys which allow characters to be inserted or deleted in the text on the screen. They can also have a key which will erase the complete text from the screen. There are also keys that can be programmed to accomplish a certain task. These keys are called function keys. For example, a function key might be programmed for use as a firing arm in a game. Function keys are very convenient and useful when rapid data entry is needed.

(3) BAR CODES.

A bar code is a series of vertical bars that represents numbers and alphabets. Most retail products today in U. S. A., have the Universal product code (UPC), a special set of bar code characters which have been universally accepted by retailers. These codes are printed on the labels of their products.

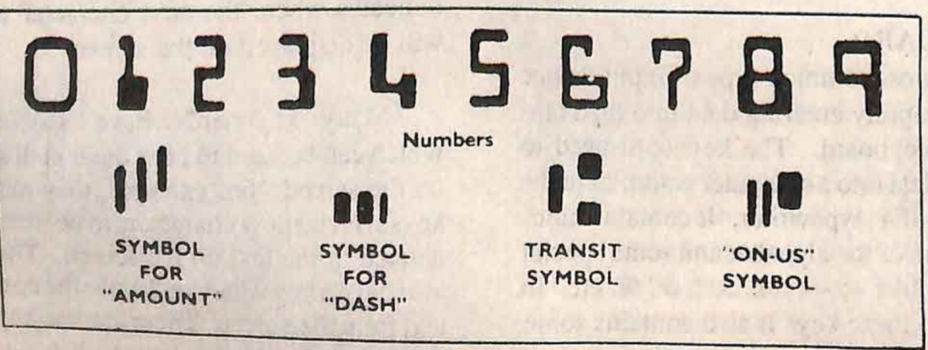
These codes are read using **Optical scanning devices.** A laser beam is emitted from the device. When the product is placed under this beam, it reads the bar code on the label and transmits this to the computer. This bar code uniquely identifies the manufacturer and his product from those of the others. After processing this code computer displays the price on the terminal.

0 28872 09330

Barcodes

MAGNETIC INK CHARACTER RECOGNITION (MICR)

The Magnetic Ink Character Recognition are used to read characters inscribed on documents with magnetic ink containing iron oxide particles. Magnetic Ink Character Recognition detects the magnetic patterns formed by them and matches them against the stored pattern of all the possible characters. Documents are rejected when the pattern is not matched. These are almost exclusively used in Banking industry to process the cheques. When MICR, is used, special characters are encoded on the cheque that identify numbers such as bank number and account number. When the cheque is to be processed, the amount of the cheque will also be encoded on the cheque by an operator at the bank. Then, when the cheque is presented to MICR it interprets the magnetized characters and these values will be stored and used to prepare bank statements at a later stage.



Magnetic Ink Character Recognition (MICR)

MICR devices can process approximately 1500 cheques per minute. In addition to the banks MICR is used in credit card companies and others which must process large volumes of data.

(4) OPTICAL MARK READERS (OMR)

An Optical Mark Reader device can read pencil marks on specially designed documents. The marks on the form can be either single vertical marks which indicate a response to a question or to fill in the specified area of the document. When these documents are presented for processing, a thin beam of light emerges from the reader and is projected on the paper. The amount of light transmitted through the mark is less as compared to the surrounding area and this leads to the sensing of the mark.

These are extensively used in scoring tests having multiple choice or objective type questions. About 10,000 documents can be processed in one hour and this enables the preparation of examination results in very short time. At the same time the errors in entering the data manually are eliminated.

OUTPUT DEVICES

(1) VISUAL DISPLAY UNIT

The most popular input/output device that is used for direct access processing is the visual display unit or monitor. It consists of a key board identical to type-writer key board and a Cathode Ray Tube (CRT) screen which displays the characters. This device is popular as it is not very expensive, easy to operate and the input/output can be seen on the screen. The size of the screen varies from 8" to 20" and can display 20 to 30 lines. The number of characters per line vary from 40 to 120. The set of characters that can be displayed on the screen are alphabets—both upper and lower case, numbers and special characters. The screen displays an indicator known as cursor. The position of the cursor indicates the position where the next character will be displayed on the screen. It keeps moving along the horizontal line as the characters are displayed. and can be moved to any desired position on the screen. We can make corrections by deleting the characters already printed on the screen or by inserting some new characters by placing the cursor at appropriate position on the screen.

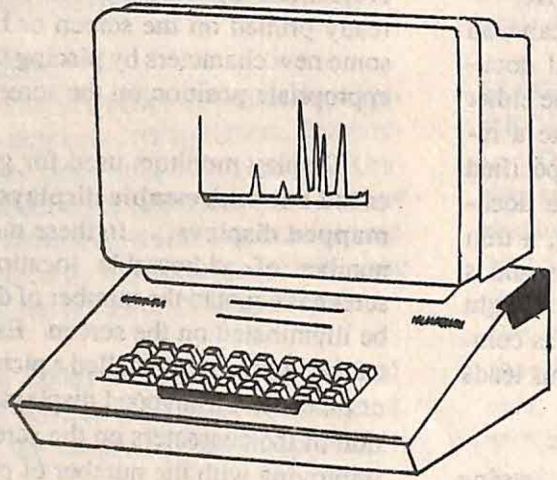
Display monitors used for graphics are called **dot-addressable displays** or **bit-mapped displays.** In these monitors, the number of addressable locations on the screen is equal to the number of dots that can be illuminated on the screen. Each dot that can be illuminated is called a picture element or pixel. In bit-mapped displays the resolution of the characters on the screen goes on improving with the number of pixels.

There are two types of monitors. These are (1) Monochrome Monitors (2) Colour Monitors.

Monochrome monitors are specially designed for use with personal computers or for use as computer terminals. These units display the characters in white or green colour on a black background. These are suitable for alphanumeric applications such as word processing.

Colour monitors are mostly used with mini computers or large computers. The colour monitors are dot-addressable display. There are two types of colour monitors-composite video monitors and RBG monitors. Both monitors can be used for colour graphics.

In monochrome monitors a single electron beam strikes the phosphorous screen causing the chosen dot to light. If the characters are green on a black background then the phosphors emit a green light when activated. In order to get colour on CRT monitor, three phosphor dots are required for each pixel. These dots are red, blue and green.



VISUAL DISPLAY UNIT

The electron beam turns on the desired colour phosphors within the pixel in order to generate an image. In a simple colour monitor eight colours can be generated. By varying the intensity of the electron beam striking the phosphors, many more colours can be generated. A composite video monitor uses a single electron signal to turn on the colour phosphors in the pixel. Whereas an RBG monitor requires three signals, one for each colour to turn on the required phosphors. An RBG monitor produces a much clear display than the composite video monitor. The colour monitors are widely used for graphic outputs where different parts of a graph require different colours.

(2) PRINTERS

The primary use of a computer is to produce useful information. One of the most widely used form of getting this information by the user is printing the information on a sheet of paper, which is also called **hard copy.** All computer printers can be classified into two categories. They are (i) Impact Printers and (ii) Non-impact Printers.

Impact printing devices transfer the character on to paper by striking the paper, ribbon and character together in two ways. The first technique is **front striking.** In front striking method the printing mechanism contains a solid character or a series of pins which can form a character which strikes a ribbon against paper to form an image. The second technique is **hammer striking** in which the ribbon and paper are struck against the character by the hammer to form the image on the paper.

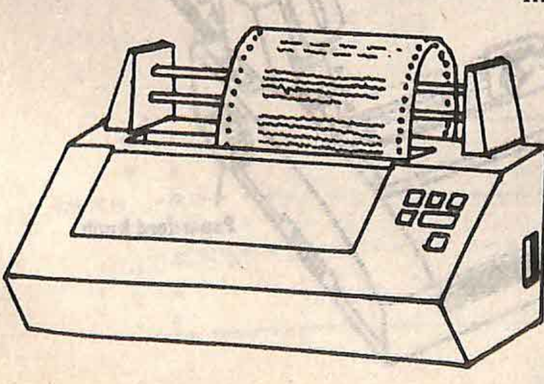
In, Non-impact Printing, printing will be done without having characters striking against paper. An example of Non-impact Printer is **Ink Jet Printer.** In this type of printer small drops of ink are used to form a character on a page.

Impact printing is relatively noisy and we can get multiple copies using carbon paper. But in non-impact printing it is not possible to use carbon paper to get multiple copies. However, they are very quiet and very fast. Printers can also be classified according to their speeds. The speed of printer is determined by the speed at which characters/lines can be printed on the page in unit time. According to speed, printers are classified as low speed printers, high speed printers and very high speed printers. The rate of printing for low speed printers is in terms of the number of characters that can be printed in one second. Low speed printers can print 15 to 600 characters per second. Most low speed printers are serial printers, i.e., the characters are printed one at a time. The rate of printing for high speed printers is in terms of the number of lines per minute that can be printed. High speed printers can print 300 to 2500 lines per minute. These printers are also called line printers as they print entire line at a time. The printers which can print more than 3000 lines per minute are called very high speed printers. Some of the very high speed printers can print 20,000 lines per minute.

LINE PRINTERS

There are two types of line printers. They are the drum printers and chain printers.

The drum printer, consists of a cylindrical drum. The characters to be printed are embossed on its surface. One complete set of characters is embossed for each and every print position on a line. The characters to be printed on one line are transmitted by the computer to the printer. The printer drum is rotated at a very high speed. A set of print hammers, one for each character of a line, are mounted in front of the drum. A carbon ribbon and paper are interposed between the drum and hammers. A character is printed by striking a hammer against the embossed character on the surface.



Line Printer

As the drum rotates, the hammer waits, and is activated when the character to be printed reaches that position. Printer drums are expensive and cannot be changed often.

The chain printer contains numbers, letters of the alphabet and special characters on a rotating chain. The chain consists of a series of type slugs which contain the character set. The character set on the type slugs is repeated two or more times on the chain. The chain rotates at a very high speed. Each possible print position has a hammer that can strike against the back of the paper. This forces the paper and ribbon against type slug to get the impression of the character on the paper. As the chain rotates the hammer strikes when the character to be printed is in correct position.

The chain printer is very reliable and produces good quality printing. The main advantage is that chains can be replaced very easily.

Line printers are generally used for heavy printing applications and can operate continuously for a few hours.

SERIAL PRINTERS

The serial printers print one character at a time with the print head moving across the line. They are similar to the type writers. The most popular serial printer is dot-matrix printer.

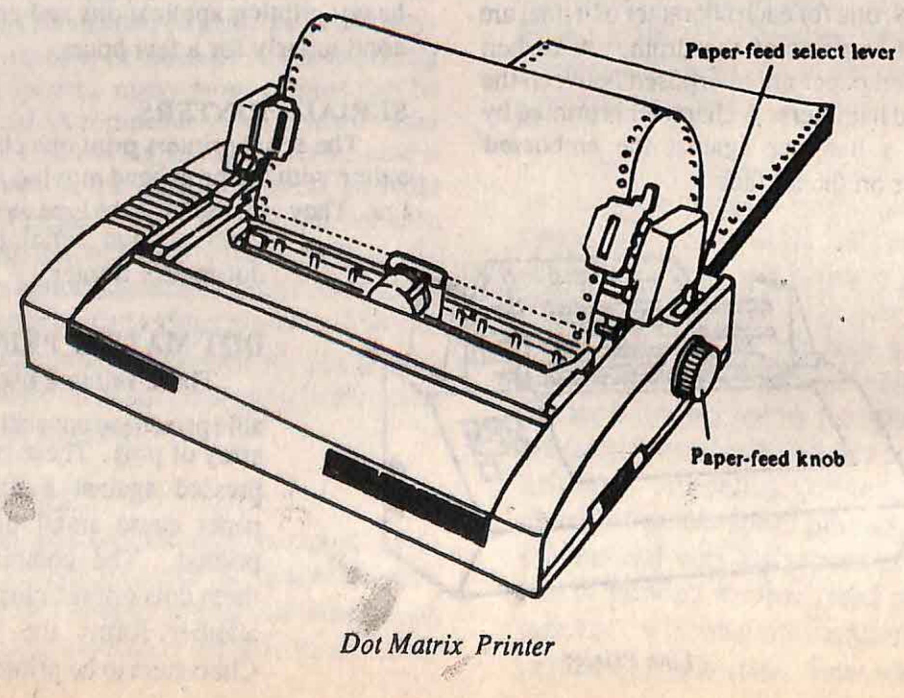
DOT MATRIX PRINTER

These printers use a movable print head consisting of 7×9 array of pins. These pins when pressed against a ribbon and paper cause small dots to be printed. The combination of these dots printed closely to one another forms the character. Characters to be printed are sent one character at a time from the memory to the printer and the printer activates appropriate pins in the print head. These pins are moved forward to form a character and they hit the carbon ribbon in front of the paper there by printing that character. Many dot matrix printers print **bidirectionally**; that is, the head prints while moving from left to right and also while moving from right to left. Printing bidirectionally almost doubles the number of characters that can be printed in a given period of time. Dot matrix printers are also able to print graphics in colours. For printing colours these printers use red, green, and blue coloured ribbons. By repeated printing and repositioning of the paper, printer head and ribbon colour output is obtained. Most dot matrix printers print either 80 or 132 characters on a line using standard paper sizes $8\frac{1}{2} \times 11$ inches and 11×14 inches respectively.

The only disadvantage of these printers is that they cannot be used continuously for more than an hour or so and are thus called **light duty printers.**

LETTER QUALITY PRINTERS

As a character produced by a dot matrix printer is made up of dots, the appearance of the printed output is not very good. When high quality output is required, for example in a text or in a correspondance letter, solid type printers are used. These solid type printers are called letter quality printers. One such letter quality printer which is most often used is the **daisy wheel printer.** The daisy wheel head resembles the structure of a flower with long thin petals. Each petal has a character on the tip. The daisy wheel is mounted on a carriage which can move from left to right and back. When printing occurs the daisy wheel rotates so that the character to be printed is in printing position, characters to be printed are sent serially to the printer. The carriage positions itself at the



Dot Matrix Printer

print position and the daisy wheel rotates so that the character to be printed appears in front of the ribbon. A hammer then strikes and forces the selected character against the ribbon and paper, causing the character to be printed. Because of the time required to rotate the daisy wheel, the daisy wheel printer is normally slower than a dot matrix printer. The printing speed varies from 20 to 60 characters per second.

The print quality of a daisy wheel is very high and the daisy wheel can be replaced very easily. Daisy wheels of a variety of type faces, such as Roman, Italic, Gothic are available.

(3) PLOTTERS

Though some dot matrix printers are capable of creating drawings, when hard copy drawings of good quality are required, computer plotters are normally used.

A computer plotter is an output device that can create drawings, charts, diagrams and similar types of graphic output. In addition, it can also print letters of alphabet and numbers to identify selected portions of the output. There are two types of plotters, flatbed plotters and drum plotters.

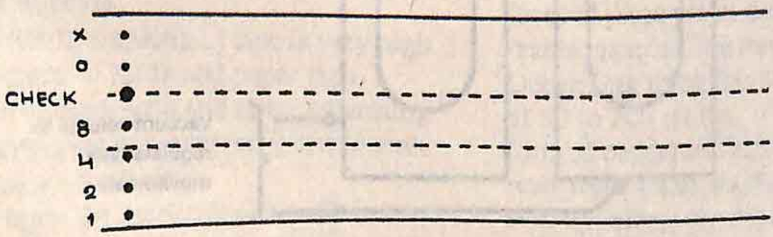
A **flatbed plotter** has a stationary horizontal plotting surface on which paper is fixed. The pen is mounted on a carriage which can move either in X or Y direction. The pen can also be moved up or down. A graph plotting computer program is used to move the pen to trace the desired graph.

A **drum plotter** uses a rotating drum over which drawing pens are mounted. The pens can move left and right as the drum rotates. The paper on which the graph is to be drawn is mounted on the rotating drum. The drum can rotate in either clockwise or anti clockwise direction under the control of the instructions sent by the computer. The movements of the pens and drum are controlled by the graph plotting program. The program can thus draw various graphs. An advantage of the drum plotter is that the length of the plot is unlimited, since roll paper can be used.

SECONDARY STORAGE DEVICES

(1) PAPER TAPE

Paper tape which is 1" in width has 10 columns in one inch length of tape. The data is recorded by punching round holes across its width. Paper tape has 8 channels. The



Paper Tape

bottom four channels represent the 8, 4, 2 and 1 numeric bit positions. The channels X and O are called **zone channels.** The presence of a hole in any column represents the bit 1. The data is recorded in the form of binary strings.

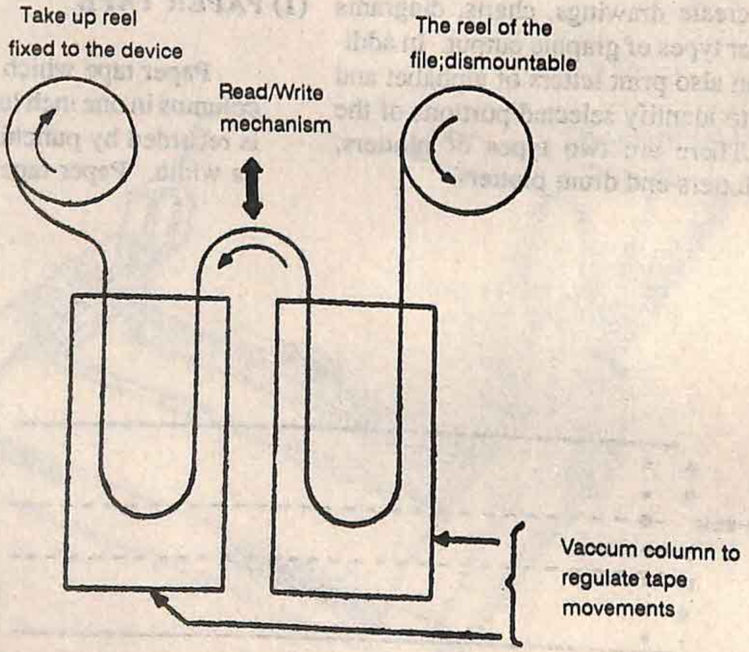
The drawbacks of Paper tapes are :

(1) It is difficult to delete or add data.
(2) They are easily torn and mutilated.
(3) The input speed is very slow.
(4) It is non-reusable.

(2) MAGNETIC TAPES

The magnetic tapes used in computer are similar to those used in audio-cassette recorders. Magnetic tape is one of the widely used form of external storage. The width of the tape is normally in the range of 1.25 cm to 2.5 cm. It is made of plastic and coated on one side with magnetic metallic oxide. Data is recorded in the form of magnetic spots. These magnetic spots are recorded into a series of horizontal rows called **channels** or **tracks.**The code for various characters and numbers are in the form of binary strings. Depending upon the codes used for recording the data, the number of tracks on the tape are either 7 or 9. If the tape is of nine tracks, then eight tracks are used to store the bits representing the character. The ninth is a parity track used to ensure that bits are correctly written to form a character by either on or off as required to ensure an odd number of 'on' bits for each character.

Data is stored on a magnetic tape in blocks of characters. A block is the unit of transfer between the magnetic tape and the main storage of the computer. These blocks are separated by gaps called **interblock gaps.** This gap is used to allow for the starting and

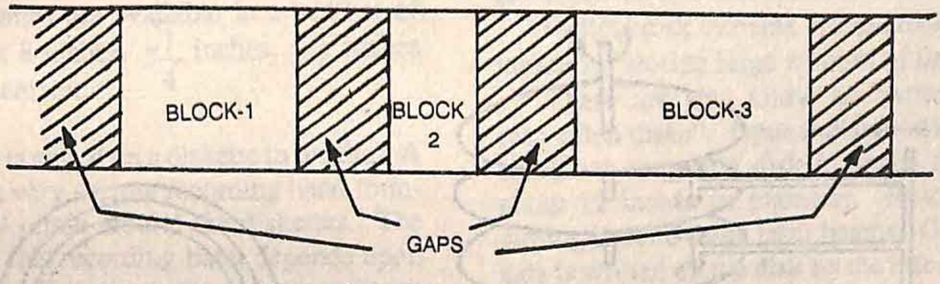


Schematic of a Magnetic Tape Unit.

stopping of the tape during read/write operations. Data on the magnetic tape is stored sequentially, i.e., one after the other and reading is also sequential. Magnetic tape is primarily used for backup purposes in large computer installations.

OPERATION OF THE TAPE DRIVE

The tape whose length varies from 250 metres to 900 metres is spooled on a reel. The feed reel is loaded on one reel holder and a plastic leader is threaded along the prescribed path to the take up reel. First the tape passes through a reservoir and then to the read/write heads. From there it passes through the other reservoir and then to the take up reel. When data is transferred from the internal storage to the tape deck, the read/write heads are activated according to the data presented and the corresponding data codes are recorded on tape. The reservoir connected to the take up reel receives the tape from the read/write head and then it is wound on the take up reel.



Recording of Data on Magnetic Tape.

Advantages :

(1) It is very easy to handle as compared to deck of cards.

(2) The storage capacity of tape is very high compared to cards and paper tape.

(3) It can be used again and again by erasing the information written on it and recording new information.

(4) The tapes are very versatile. They can transfer data to memory at high speed, i.e., about 10,000 to 3,00,000 characters per second.

(5) The price of tapes is also very cheap compared to punched cards.

Disadvantages :

(1) Magnetic Tape allows only sequential retrieval, i.e., in order to reach a particular data item recorded in the middle, it is necessary to roll through all the previous blocks. Because of this process a lot of time is wasted.

(2) Some data may be lost when the tape is worn out.

(3) MAGNETIC DISKS

Magnetic disks are electromagnetic devices similar to magnetic tapes. In addition to sequential access, disk allows for direct access, i.e., any particular piece of information can be read or written without having to access other data. This allows faster retrieval of information.

Disks are circular plates of 75 cm to 375 cm in diameter. Like a phonograph record disks spin about their centre. Both the surfaces of these plates are coated with magnetizable material like magnetic metallic oxide. Depending upon the size each disk consists of 50 to 200 tracks. These tracks are in the form of concentric circles. Data is written or read from these tracks. Data are recorded serially along a single track. Each disk surface is divided into a fixed number of sectors. Though the length of each track is different, same number of characters are stored in each track. So the inner tracks are densely packed with characters than the outer ones. A magnetic disk unit consists of 5 to 50 disks mounted on a common spindle about 1/2" apart. The outer surfaces, i.e., the top surface and the bottom surface are not used for recording, i.e., if a disk pack contains 6 disks then there will be 10 usable surfaces. Each surface is provided with one arm which contains read/write heads to read or write information on that surface. The storage segment is a particular sector of a specific track of a disk. In general disks store 512 bytes per sector. Hence the total capacity of the disk pack containing 50 sectors per track, 400 tracks per surface and 10 surfaces is 512 × 50 × 400 × 10 = 102.4 million bytes. Information can be accessed directly as each storage segment has a unique identity. This identity consists of surface number, track number and sector number. The sector method of storing and retrieving information is sometimes called as fixed block architecture.

Advantages :

(1) Because of the direct access, the access time of disk is a fraction of access time required by the tape.

(2) Disk storage is more durable.

(3) It is a very efficient I/O medium.

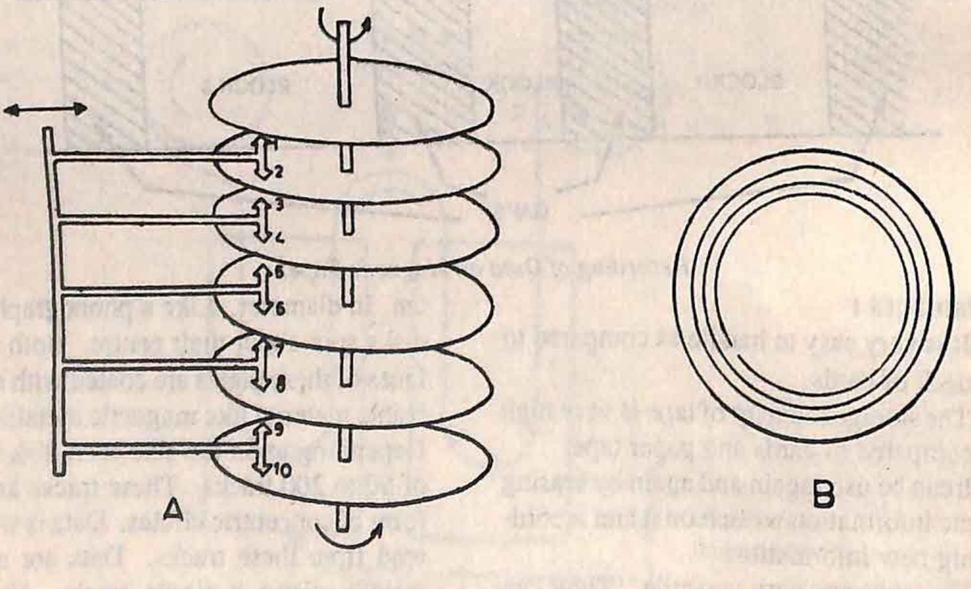
Disadvantages :

(1) Magnetic Disks are much costlier than tapes.

(2) Because of the heavy weight it is more difficult to replace a disk pack than a reel of tape.

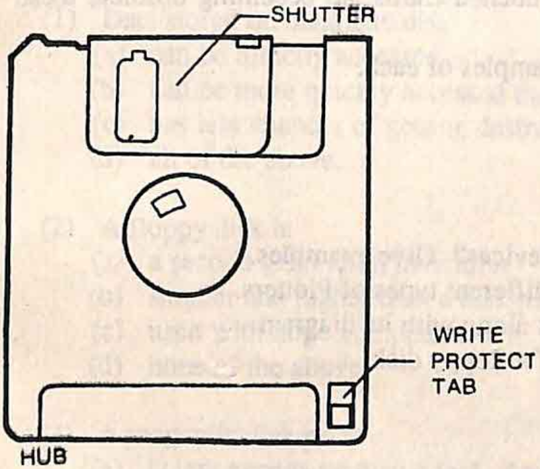
(4) FLOPPY DISKS

A floppy disk or diskette consists of a circular piece of thin plastic material coated with an oxide material similar to that used on magnetic tapes. The circular piece of plastic is enclosed in a square protective jacket with a cut out called recording window so that the magnetic surface is exposed. The magnetic surface of the diskette which is exposed through the window in the jacket allows data



(A) Schematic of a disk unit with 6 plates, 10 recording surfaces, and movable access arm.
(B) The concentric circles (tracks) on a surface in which data is stored.

to be stored or read. The large hole in the diskette is used to mount the diskette into a hardware device called floppy drive. The small hole is used to indicate the floppy drive where to store data.



Floppy Disk

Diskettes are available in a number of sizes, like 8-inches, $5\frac{1}{4}$ inches, $3\frac{1}{2}$ inches etc.,in diameter.

Data is stored on a diskette in **tracks.** A track is a very narrow recording band forming a full circle around the diskettes. The width of this recording band depends upon the number of tracks on the diskette and each track is separated by a narrow gap. The number of tracks on a diskette depends on the floppy drive. The two most commonly found configurations are 40 tracks and 80 tracks.

For read/write operations on a diskette, the diskette is inserted in a disk drive. The disk within the protective covering is rotated at about 300 revolutions per minute. To read or write data, a read/write head in the floppy drive rests on the surface of the rotating diskette, sensing or generating electronic impulses representing bits to be read or recorded. To read the data from different tracks, the read/write head moves from track to track.

Advantages

(1) Error correction is easier than in cards and paper tapes.
(2) Easily transportable (one diskette can store information stored on 4000 cards).
(3) It can be used for both input and output.
(4) Relatively cheap.
(5) It can be used many times.

Disadvantages:

(1) Floppy disks have to be handled very carefully.
(2) It can be damaged easily by adverse environmental conditions like heat, humidity etc.

(5) OPTICAL DISK SYSTEMS

Optical disk systems are relatively new means for storing large amount of data.

These are also know as "write once, read-often disks". Optical disk systems are similar to magnetic disk in shape and are about 12 inches in diameter. Reading or writing is done using laser beams. Once the data is written on the disk by the laser beam, the data cannot be erased or changed.

Large amounts of data can be stored on an optical disk. Approximately one billion characters can be stored on a 12-inch disk. The number of tracks per inch in a magnetic disk never exceeds 96 but in Optical disk the number of tracks per inch is around 15,000. Thus it can be seen how large amounts of data can be stored on the Optical disk. Optical disks are currently used for storing the data which will remain constant and does not need to be changed. One such type of data, for example is archival data.

EXERCISES

ESSAY TYPE QUESTIONS

1. What do MICR and OMR denote? Explain their function briefly.
2. State the advantages of magnetic disk storage over other storage techniques.
3. Distinguish among Character, Line and Page Printers.
4. Explain the advantage of a Daisy Wheel printer as compared to a Dot Matrix Printer.
5. Write a short note on Punched Cards. "Punched Cards are becoming obsolete these days", give reasons.
6. Explain any two output devices giving examples of each.

SHORT ANSWER QUESTIONS

7. Explain the working of a Line Printer.
8. What is a VDU? What is it used for?
9. What do you mean by Input and Output devices? Give examples.
10. Write a short note on Plotters stating the different types of Plotters.
11. Give a brief description of the floppy disk along with its diagram.
12. State the advantages and disadvantages of a floppy disk.

OBJECTIVE TYPE QUESTIONS

13. Fill in the blanks :

(1) .... devices are used to feed the data to the computer.

(2) The Input devices convert digits, letters and other information to the internal .... code used by the computer.

(3) A punched card consists of .... rows and .... columns.

(4) .... can directly read the printed characters and convert them into the codes understood by the computer.

(5) A line printer can print.................. to .... lines per minute.

(6) .... printers print one character at a time.

(7) .... is the most common input device.

(8) IKB in computer terminology means ....

(9) ....is an input device through which one can communicate with the computer.

(10) The keys in a keyboard are arranged in the same manner as in a ....

(11) The rows 1 to 9 of a punched card are called ....

(12) The information punched on the cards is fed to the computer through an input device called.........................

(13) Two of the most widely used special purpose keys of a key board are .... keys.

(14) Keys that can be programmed to accomplish a certain task are called..............

(15) A......................... is a series of vertical bars that represents numbers and alphabets.

(16) An.................... device can read pencil marks on specially designed documents.

(17) The two types of monitors are.................... and.............. monitors.

(18) The two types of printers are.................. printers and .... printers.

(19) A .... is an output device that can create drawings, charts, graphs etc.

(20) The width of magnetic tape is normally in the range of .... to ....

(21) The row 11 and 12 of a punched card are called ....

(22) The row 0 of a punched card is used for .... and ....

(23) Optical disks are also known as .... disks.

14. Complete the sentence by choosing one of the alternatives given.

(1) Data stored on magnetic disk
   - (a) can be directly accessed.
   - (b) can be more quickly accessed than the magnetic tape
   - (c) has less chances of getting destroyed than magnetic tape
   - (d) all of the above.

(2) A floppy disk is
   - (a) a second generation invention
   - (b) smaller and higher than a disk pack
   - (c) used with large computers
   - (d) none of the above.

(3) A magnetic disk pack
   - (a) is less expensive than a reel of magnetic tape of equal storage capacity.
   - (b) is lighter than magnetic tape
   - (c) has much large capacity than a magnetic reel
   - (d) all of the above.

(4) Multiple choice examination answer sheets are evaluated by
   - (a) Magnetic tape reader
   - (b) Magnetic ink character reader
   - (c) Optical character reader
   - (d) Optical mark reader.

(5) Which of the following are secondary storage devices ?
   - (a) Magnetic disk
   - (b) Plotter
   - (c) MICR
   - (d) OMR.

□□

6

PROBLEM SOLVING

Many people refer to this era in history as the age of the computer. It is now recognized that an understanding of the computer, how it works, how it processes data to produce useful information and how it can be used as tool in home, school, and our work environment is a necessary part of the general education for all who live and work in this information processing age.

Computers are popularly, though mistakenly, perceived to be electronic brains having the attributes of intelligent beings. They only help human beings to solve problems and in decision making by processing information given to them. They do exactly what they are told and just do not make any mistake. They have no common sense. The type of problems that computers can solve are confined only by limitations on human creativity and imagination. Any task that can be carried out systematically, using a precise step by step method can be performed by a computer.

Every problem has some given information and some result is to be obtained. The question is how to process the given information or data so that the required result can be obtained, i.e., what procedure should be applied to the given information so that we can get the required result. The development of a proper procedure is called **algorithm** for problem solving. The following examples explain the concept of algorithm which should be straight forward and self explanatory.

Example 1.

Obtain the sum of two numbers.

These numbers may have any value. So let us call them X and Y. Suppose the information about X and Y is given.

```
Step 1 :  Read the values of X and Y.
Step 2 :  Add X and Y and call it Z.
Step 3 :  Write the value of Z on the
          paper.
Step 4 :  Stop.
```

Suppose in the above procedure step 3 is missing. Follow the steps starting from step 1 and see what will happen.

Example 2.

Obtain the percentage of marks obtained by the students in a given examination.

In this problem, the information is given about the marks obtained and the maximum marks out of which these marks have been obtained. The required result is the percentage of marks.

```
Step 1 :  Read student Name, Marks ob-
          tained and Max. Marks.
Step 2 :  Divide the marks obtained by
          Max. Marks and call it A.
Step 3 :  Multiply A by 100 to get per-
          centage.
Step 4 :  Write the Name and percentage
Step 5 :  If you want to  continue go to
          step 1.Otherwise go to step 6.
Step 6 :  Stop.
```

Thus, we see that to solve any problem a procedure is needed. If the problem is to be solved by a computer, then there should not be any ambiguity in the procedure. Everything in the procedure should be clearly and properly defined such that if that procedure is to be followed even by a man who does not want to apply his brain to think, he should be able to solve the problem or he should be able to get the required information. To illustrate this method, let us take some simple examples from our day-to-day life and try to obtain a step-by-step procedure to solve them.

Example 3.

Suppose you want to telephone your friend to give him/her some information. In the following procedure, it is assumed that you know the telephone number.

Step 1 : Take the receiver in your hand and dial the telephone number of the person with whom you want to talk.
Step 2 : If it starts ringing, go to step 3 otherwise go to step 7.
Step 3 : If someone speaks from the other side, verify the number you have dialed. If no one speaks then go to step 8.
Step 4 : If the number is correct, talk to your friend or get information about him. If the number is wrong goto step 10.
Step 5 : Keep the receiver on the telephone.
Step 6 : Stop.
Step 7 : The telephone on the other side seems to be engaged. Keep the receiver on the telephone. Wait for a minute and go to step 1.
Step 8 : The ring is going but nobody is picking up the phone from the other side.
Step 9 : No one seems to be at home. Go to step 5.
Step 10 : It seems you have got the wrong number. Say sorry and keep the receiver on telephone. Go to step 1.

This procedure for telephoning your friend will work in most of the cases. But if someone does not know the telephone number, a query occurs in Step 1, i.e., step 1 cannot be performed, as in step 1 the telephone number is needed and that number is not known. So there is a need to revise this procedure to handle these type of situations. This is how a procedure is improved to handle more general problems. The main aim of a procedure should be such that no thinking be required while performing the steps of the procedures as these procedures are to be performed by the computer which cannot think but can faithfully follow the instructions. The procedure should also work in as many situations as possible like in example 2, where the same procedure will be repeated a number of times depending upon the number of students.

The procedures are performed step-by-step starting from step 1, i.e., it performs first step 1, then step 2 and so on. At any stage-depending upon the situation, if one wants to perform a step other than the step immediately following it then the statement like "go to" is used. Then from the step which is currently being performed one will move to the step mentioned in go to statement ignoring all the steps in between. For example, in the above procedure in step 3, if there is no response to the ring then it is meaningless to go to step 4.

Now let us consider a very common activity required everday, which is washing ones face.

Step 1 : Go to the wash basin.
Step 2 : Open tap.
Step 3 : Wash face with water.
Step 4 : Close the tap.
Step 5 : Take soap from soap case.
Step 6 : Apply soap on hands.
Step 7 : Keep soap on the soap case.
Step 8 : Massage the soap on the face. with your hands.
Step 9 : Open the tap.
Step 10 : Wash the face.
Step 11 : Close the tap.
Step 12 : Take the towel and dry up the face with it.

Step 13 : Keep the towel on the stand.
Step 14 : Stop.

We note that the above procedure can be performed by a child who understands these simple steps. This procedure can be modified to include the situation when the soap is not there in the soap case etc. Thus we have seen that to use computers for problem solving, the step-by-step procedure is necessary. These steps have to be serialwise i.e., one after another, as the computer operates serially.

From the above examples we conclude :

1. Procedures contain inputs.
2. A set of steps are required to process inputs to produce an output.
3. Procedures are more precise and unambiguous. There is little chance for misinterpretation.
4. The number of instructions used in the procedure are finite.

We will now see when a step-by-step procedure can be called **Algorithm.** The origin of the word algorithm comes from the name of a famous Arab mathematician Abu Jafar Mohammed Ibn Musa al-Khowarizmi (literally meaning father of Jafar Mohammed, son of Moses, native of al-khowarizmi). He wrote a famous book "Rules of Restoration and Reduction". The last part of his name al-khowarizmi was corrupted to algorithm. An algorithm can be defined as a finite sequence of instructions with the following characteristics:

(a) An algorithm starts with input instructions. These inputs are processed by the subsequent instructions in the algorithm.
(b) The rules specified in the algorithm must be precise and unambiguous.
(c) Each instruction should be basic i.e.,it should be carried out by a person in finite number of steps.
(d) The total time to carry out all the steps in the algorithm must be finite.
(e) Finally algorithm must produce an output.
(f) Each algorithm must have an end statement.

EXERCISES

ESSAY TYPE QUESTIONS

1. You have lost the telephone number of your friend. Write a step-by-step procedure to locate his/her telephone number from the directory and then make a telephone call.
2. Define an algorithm. What are its characteristics?
3. Write a step-by-step procedure to find geometric mean of four nos. (Geometric mean is found by multipyling all four numbers and dividing the result by 4).
4. Write a step-by-step procedure for going to the hotel to meet your friend.

SHORT ANSWER QUESTIONS

5. Write a step-by-step procedure for finding largest of 3 numbers.
6. Write a step-by-step procedure to find percentage of marks obtained by students in the given examination. You are given the marks obtained and Maximum Marks out of which marks have been obtained.
7. Write a step-by-step procedure for finding product of two numbers.

OBJECTIVE TYPE QUESTIONS

8. Fill in the blanks :
   (1) The development of a proper procedure to solve a given problem is called ....
   (2) If a problem is to be solved by a computer, then the procedures should be a .... procedure.
   (3) Every problem has some information given and some .... is to be obtained.
   (4) A sequence of instructions for solving a given problem is called ....
   (5) Each algorithm must have a .... statement.

9. Arrange the following steps of crossing the road in correct order.
   STEP-1 Look at your right-hand-side.
   STEP-2 Cross the road
   STEP-3 Go to the edge of the road
   STEP-4 If there is any vehicle on the road go to step 1, otherwise go to step 5.
   STEP-5 Cross half of the road and stop in the middle.
   STEP-6 If there is any vehicle on the road go to step 7 otherwise go to step 2.
   STEP-7 Look at your left hand side.
   STEP-8 STOP.

10. Complete the missing steps in the following step-by-step procedure to find the average marks of N students in 4 subjects.
   STEP-1 Set a counter S to 0 i.e.,S = 0
   STEP-2 Set another counter I to 0 i.e.,I = 0
   STEP-3 ....
   STEP-4 Increase the counter I by 1 i.e.,I = S + 1
   STEP-5 ....
   STEP-6 Check if counter I is equal to total number of students i.e. N. If not then go to step 4.
   STEP-7 ....
   STEP-8 ....
   STEP-9 END

☐☐

7

FLOW CHARTS AND SIMPLE FOUR-LINE PROGRAMS

We have seen that to solve a problem using a computer, it is necessary to develop a step-by-step procedure. These steps should be unambiguous and serial, that is one after another such that the calculations of the current step should not be dependent on the values provided or calculated in the following steps. To solve any problem on a computer, the following steps are required .

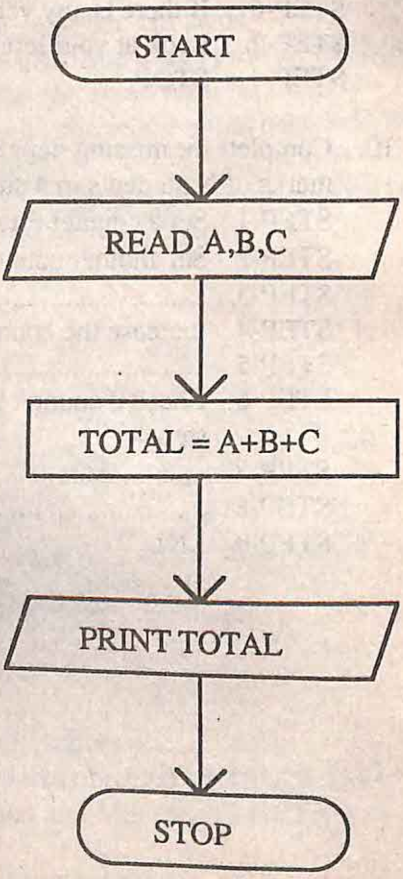
- (i) Statement of the problem.
- (ii) A well defined procedure to solve the problem.
- (iii) Coding the procedure into a particular language.
- (iv) Testing and debugging the coded program.
- (v) Documentation.
- (vi) Maintainence.

The procedures we discussed in the earlier chapters can also be represented in another way called flow charts.

A flow chart is a pictorial representation of the steps involved in the procedure. It also shows the logical sequence in which the steps are to be performed. A flow chart consists of **boxes,** called **symbols** and arrows called **flow lines.** The box tells us what to do and flow line show us the next step to be followed.

In flow charts each step is put into a box to emphasize that it is an individual unit and lines indicate the flow of sequence. For example, if one wants to calculate the total marks obtained by a student in an examination consisting of three subjects A, B and C then the following procedure can be used to solve the problem.

Step 1 : Read marks in three subjects and call them A, B and C respectively.
Step 2 : Add A, B and C and call it TOTAL.



Step 3 : Write TOTAL.
Step 4 : Stop.

Once the problem has been broken into steps and precise sequence of steps is decided, then the programmer can illustrate each individual step through a flow chart.

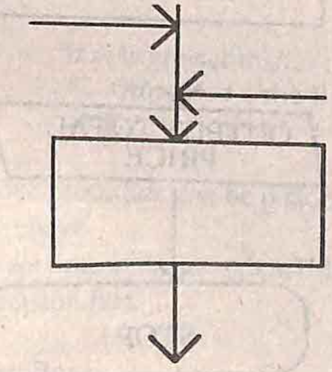
The above procedure can be shown with the help of diagrams which make the understanding of the problem easy and clear.

The above algorithm has the following steps:

Obtaining the input data, performing calculations and producing the output information. Each type of step is shown using a box of different shape. In the above flow chart you can notice the following types of boxes :-

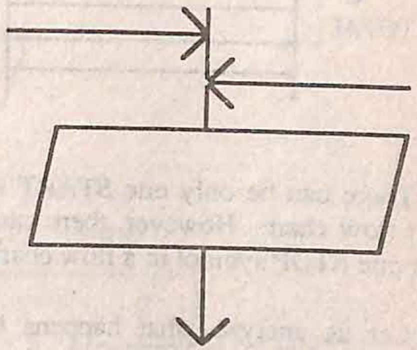
Processing box

The shape of the processing box is a rectangle. This symbol represents one or more instructions which performs a processing function of the program. Examples of processing functions are addition, subtraction, multiplication, division or moving data to storage or assigning the value to some variable as in the case of assigning the value zero to SUM i.e., SUM = 0 or X = 5, Y = 10 etc. Only one flow line can leave the processing box where as any number of lines can enter it.



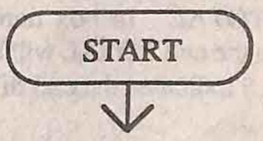
Input/Output box

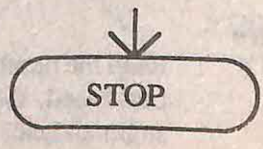
The shape of input/output box is a parallelogram. This symbol indicates any function of input/output device such as a keyboard or a printer. It makes data available for processing or displaying the result of processing on the screen. Keying in data through key-board, displaying information on the screen or printing it on the printer are some examples of input/output operations. Again only one flow line can leave this box and any number of flow lines can enter this box.



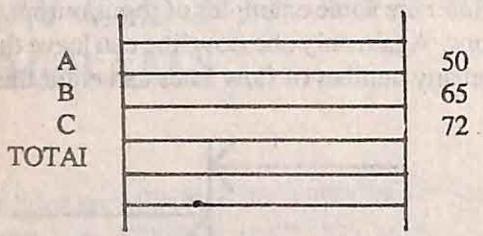
Terminal Box

This symbol is used to indicate the beginning or end of a flow chart. When this symbol is used for START, no flow lines can enter it and only one flow line can leave this box. When it is used





STOP, no flow line can leave this box and any number of flow lines can enter this box.



There can be only one START symbol in each flow chart. However, there can be more than one STOP symbol in a flow chart.

Let us analyze what happens inside the computer's memory. Computer memory can be thought to be made up of cells, where different values can be stored. Read A, B, C in box number 2 of the above example reserves memory space and stores the values in boxes named A, B and C which can have any numeric values.

Suppose values of A, B and C are 50,65 and 72 respectively. In box number 3, addition of A, B and C is done and is stored in memory cell named TOTAL, i.e., 50, 65, 72 are added and the result, that is 187, is stored in the memory cell named TOTAL. In box number 4 the value stored in the cell TOTAL will be printed and box number 5 indicates the end of the procedure.

Example 2 :

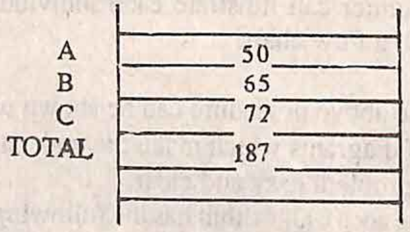
Draw a flow chart to calculate the total price of sugar purchased from a shop.

Procedure :

Step 1 : Read the number of kgs of sugar purchased, the price per kg of sugar and call them X and Y respectively.

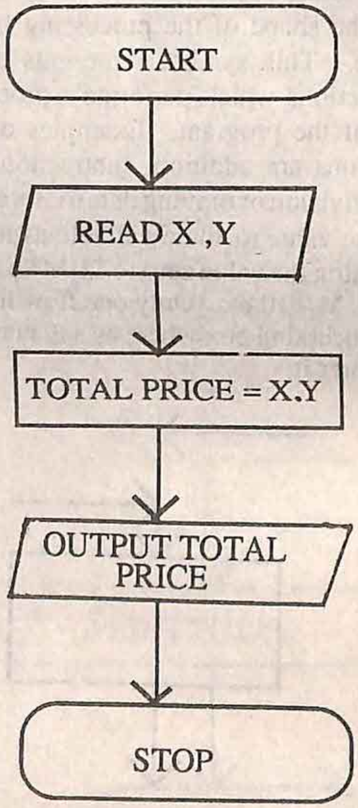
Step 2 : Multiply X and Y and call the result TOTAL PRICE.

Step 3 : Output TOTAL PRICE.



Step 4 : Stop.

The above procedure can be pictorially represented as :-

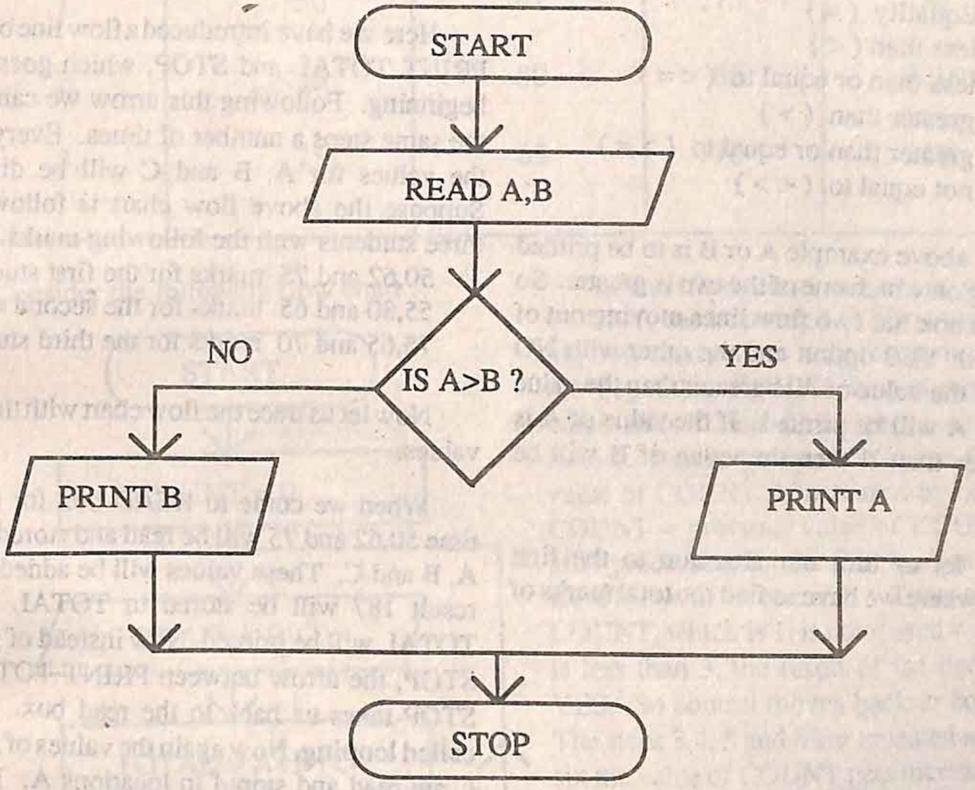


Example 3 :

Draw a flow chart for finding the largest of two numbers.

Procedure :

Step 1 : Read the two numbers and call them A and B respectively.



Step 2 : If A is greater than B then write A. Otherwise write B.
Step 3 : Stop.

The above procedure can be pictorially represented as above.

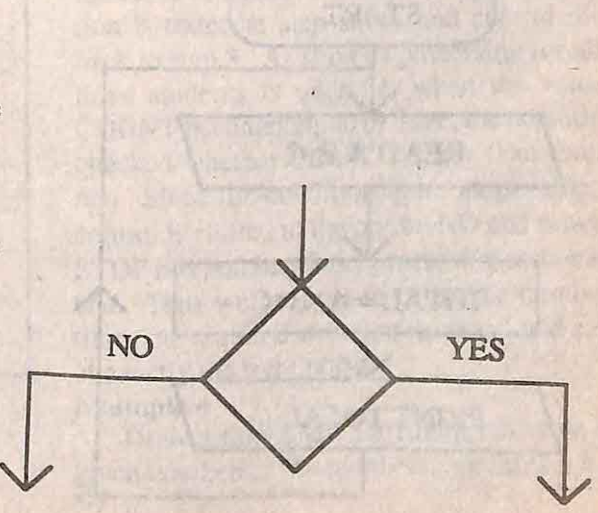
Here we find another symbol.This symbol is called Decision Box.

Decision Box

The shape of a decision box is a rhombus. This box is used when some comparison is to be performed between two quantities.

This box is also used for testing a condition.

A decision box gives rise to two alternative



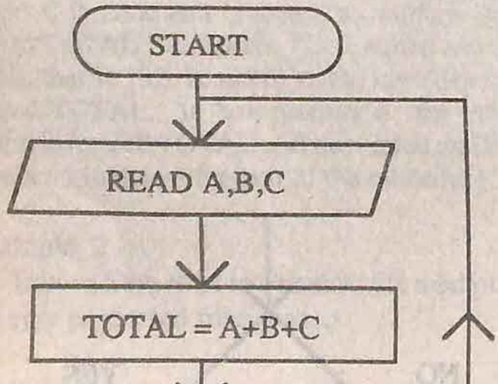
answers to the condition, that is either the condition is true or it is false.

In a decision box the following conditions can be checked :

(i) Equality (=)
(ii) less than (<)
(iii) less than or equal to (< =)
(iv) greater than (>)
(v) greater than or equal to (> =)
(vi) not equal to (< >)

In the above example A or B is to be printed depending on which one of the two is greater. So a decision box has two flow lines moving out of it, one with YES option and the other with NO option. If the value of A is greater than the value of B then A will be printed. If the value of A is not greater than B then the value of B will be printed.

Now let us turn our attention to the first problem where we have to find the total marks of one student. Now suppose we want to find the total marks for each student in the class. But this is not an efficient way. Since all the steps have to be repeated for many students, the flow chart can be modified as shown earlier .



Here we have introduced a flow line between PRINT TOTAL and STOP, which goes to the beginning. Following this arrow we can repeat the same steps a number of times. Every time, the values for A, B and C will be different. Suppose the above flow chart is followed for three students with the following marks.

50,62 and 75 marks for the first student;
55,80 and 65 marks for the second student;
75,65 and 70 marks for the third student.

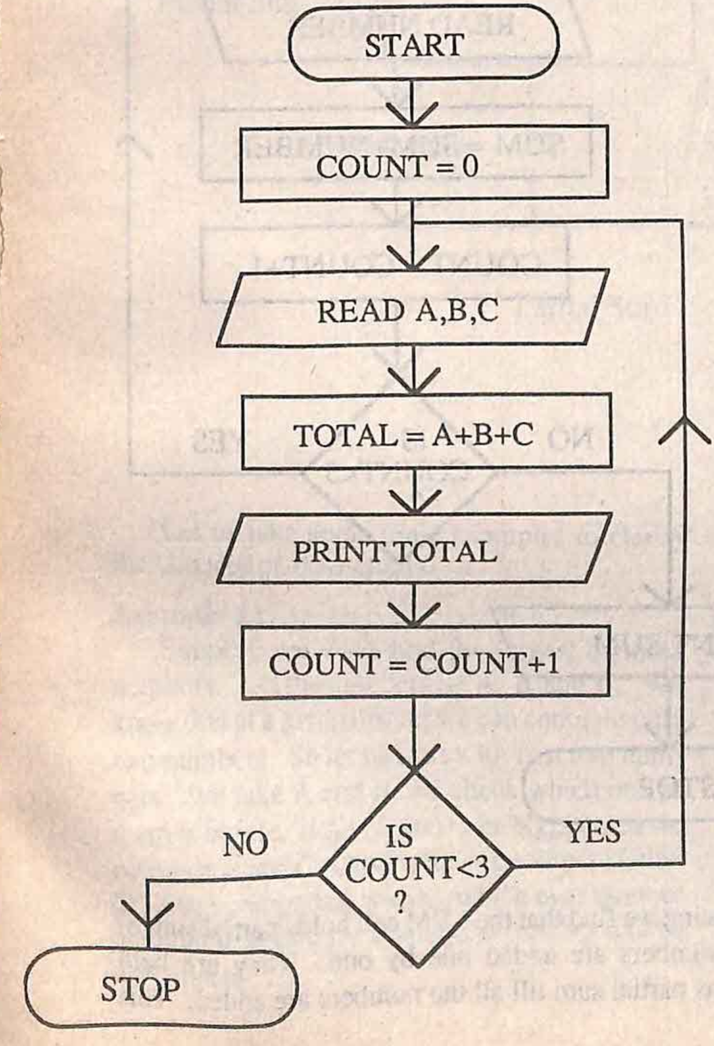
Now let us trace the flow chart with the above values.

When we come to READ box for the first time 50,62 and 75 will be read and stored in cells A, B and C. These values will be added and the result 187 will be stored in TOTAL and the TOTAL will be printed. Now instead of going to STOP, the arrow between PRINT TOTAL and STOP takes us back to the read box. **This is called looping.** Now again the values of A, B and C are read and stored in locations A, B and C overwriting the previous values. New result after adding the new values 55,80 and 65 will be stored in TOTAL overwriting the previous value in TOTAL. The change in the values of A, B and C and TOTAL can be depicted as show in the table given on following page :

In this way, by using the same memory locations and same set of steps the processing can be done for more than one student. In the above process the control will move back to box number 2 even after processing is over for all the three students and the control will never come back to STOP box. Therefore, the flow chart should be modified such that the control reaches the STOP box when the TOTAL for all the students is calculated. This can be achieved by using the concept of counting.

ITERATIONS	MEMORY CELLS			
	A	B	C	TOTAL
1.	50	62	75	187
2.	55	80	65	200
3.	75	65	70	210

Consider the following flow chart :



In this flow chart one more variable, i.e., COUNT, has been used. This is used for counting how many times the process is repeated. In the beginning COUNT is assigned a starting value zero. Then all the processing steps will follow. Before going back to the first step, the value of COUNT is increased by one, i.e., now COUNT = previous value of COUNT + 1. So after processing for the first student the value of COUNT will become 1. Then the value of COUNT, which is 1, is compared with 3 and as 1 is less than 3, the result of the decision box is YES. So control moves back to box number 3. The steps 3,4, 5 and 6 are repeated again. In step six the value of COUNT gets increased by 1, i.e., the value of COUNT is now 2. Again the condition is tested at step seven and control moves back to step 3. As soon as processing for all the three students is over i.e, when the value of COUNT becomes equal to three, the condition is checked whether COUNT is less than three or not. Since the condition is no longer true, the control is shifted to the option NO and moves to STOP box and hence the processing comes to an end. Thus we are able to count the number of times the required steps are executed and stop it at exactly the right point.

Example 4

Draw a flow chart for finding the sum of 5 given numbers. The numbers are 20, 2.5,4 and 50 .

Processing : All numbers are to be added.

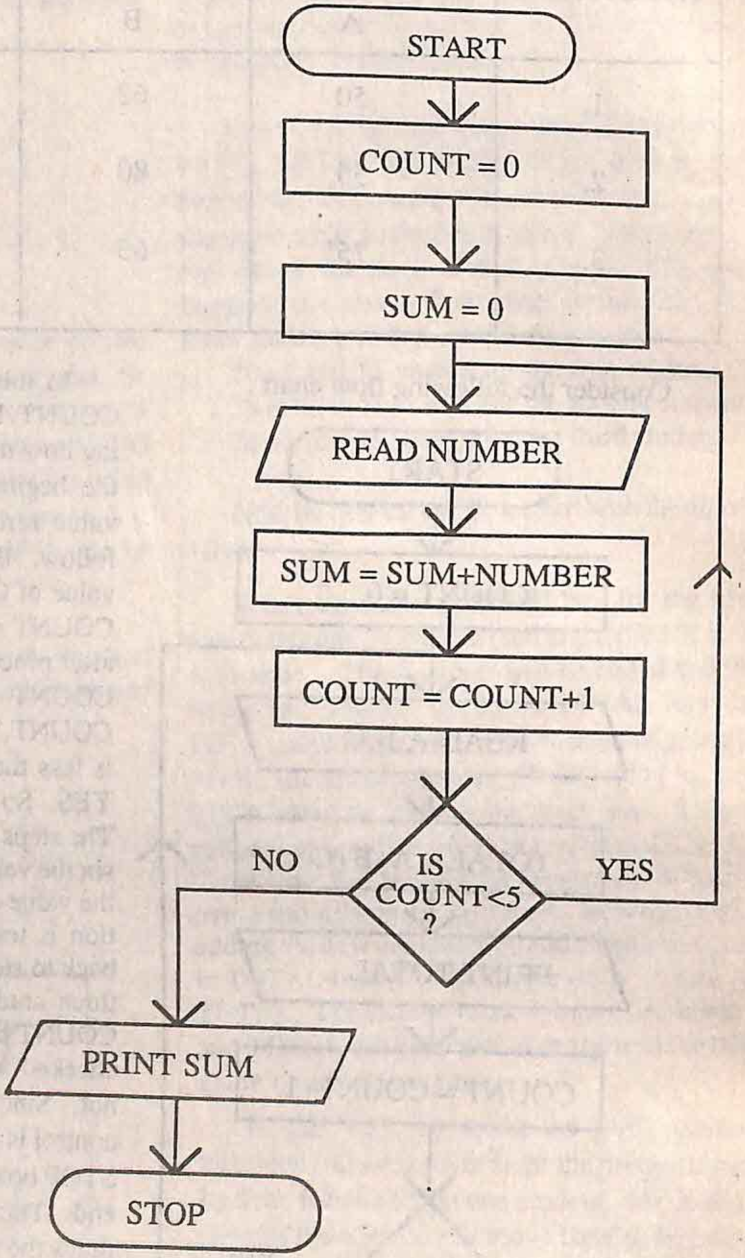
Output : Sum of all the numbers.

Let us first workout its step-by-step procedure. Since we have to read numbers more than once we require to use COUNTER.

Short Names for Memory cells are as below :

NUMBER : Value of numbers
SUM : Partial sum
COUNT : Counter for numbers.
Step 1 : Set COUNT equal to zero (since we have not started counting).
Step 2 : Set SUM equal to zero (since we have not started adding numbers).
Step 3 : Read the number.
Step 4 : Add number to SUM and call it SUM.
Step 5 : Add 1 to COUNT and call it COUNT again.
Step 6 : If COUNT < 5 then go to step 3 otherwise go to step 7.
Step 7 : Write SUM.
Step 8 : Stop.

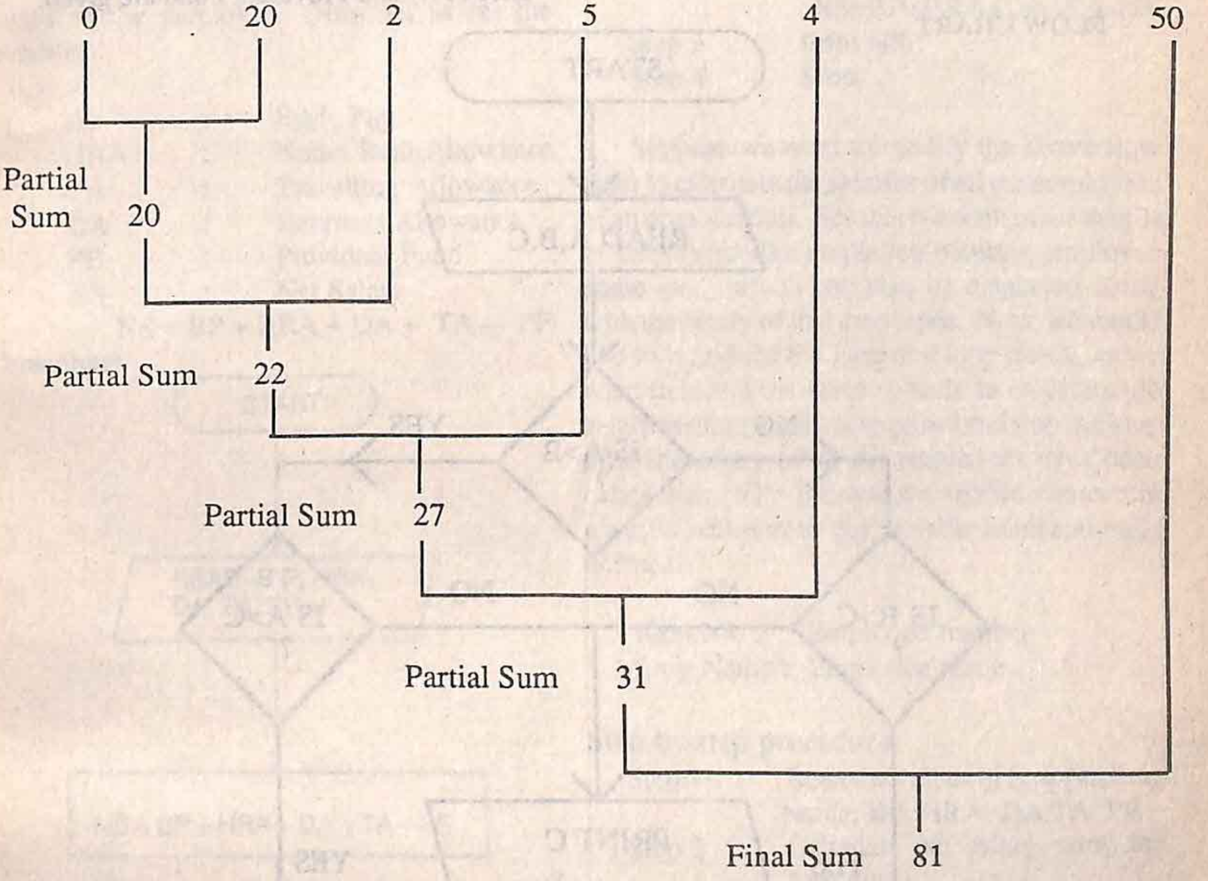
In the above algorithm we will keep on reading the numbers and adding them till all the five numbers are added, i.e., until COUNT becomes 5. In step 7, the SUM will be printed. This can be represented in the flowchart form as given below :



From tracing we find that the SUM cell holds partial sum of numbers. Numbers are added one by one. They are held temporarily as partial sum till all the numbers are added. This can be explained as in the following figure.

Short names for Memory cells ;

A : Value of the first number
B : Value of the second number
C : Value of the third number.



Let us take some more examples to clarify the concept of flow charts.

Example 1 :

Suppose we want find the largest of three numbers. Let the numbers be A, B and C. We know that at a given instant we can compare only two numbers. So let us start with first two numbers. We take A and B and check which one of them is bigger. If A is found to be bigger then we compare A and C. Otherwise, we compare B and C. After comparing with C, which ever number is found to be bigger, that bigger number will be the largest.

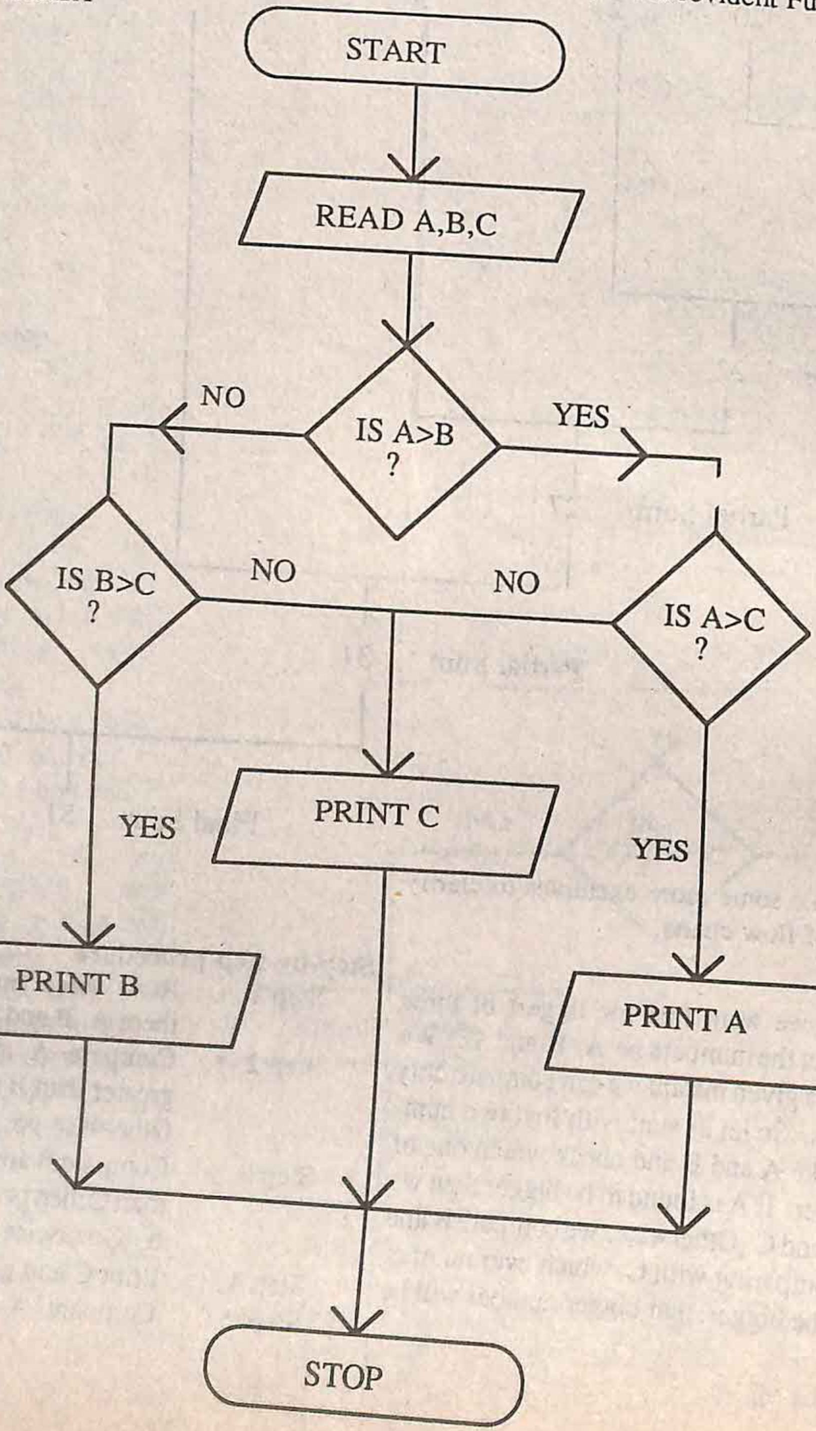
Step-by-step procedure

Step 1 : Read three numbers and call them A, B and C.
Step 2 : Compare A and B. If A is greater than B then go to Step 5. Otherwise go to step 3.
Step 3 : Compare B and C. If B is greater than C then print B and go to step 6. Otherwise go to step 4.
Step 4 : Print C and go to step 6.
Step 5 : Compare A and C. If A is

greater than C then print A and go to step 6. Otherwise go to step 4.

Step 6 : Stop.

FLOW CHART



Example 2 :

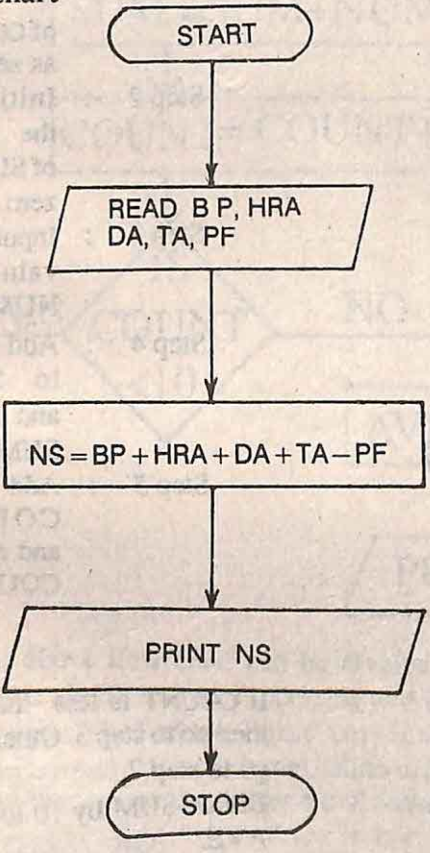
To calculate the total wages of an employee if his Basic Pay, House Rent Allowance (HRA), Dearness Allowance (DA), Travelling Allowance (TA) and Provident Fund are given.

In this particular problem we have to calculate the net salary of one employee. We also know that H.R.A., T.A. and D.A.have to be added to the Basic Pay to get the Gross Salary and P.F. is to be deducted from Gross Salary to get the Net Salary of the employee. Now let us set the variables.

BP : Basic Pay
HRA : House Rent Allowance
TA : Travelling Allowance
DA : Dearness Allowance
PF : Provident Fund
NS : Net Salary.

NS = BP + HRA + DA + TA — PF

Flow chart



Step-by-step procedure

Step 1 : Read the values of BP, HRA, DA, TA and PF.
Step 2 : Calculate net salary using the formula.
NS=BP+HRA+DA+TA-PF.
Step 3 : Print NS.
Step 4 : Stop.

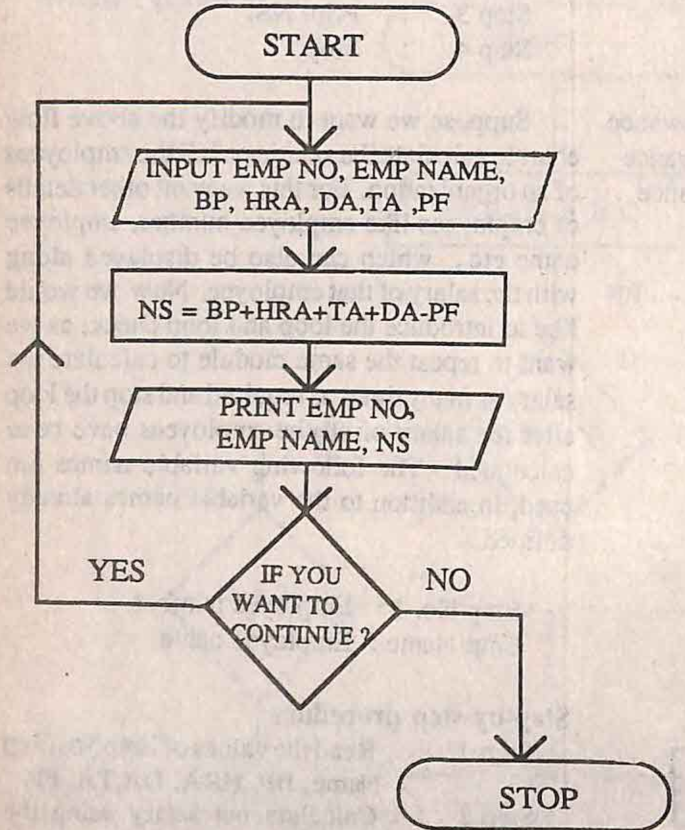
Suppose we want to modify the above flow chart to calculate the salaries of all the employees of an organization. For this we want other details of employees like employee number, employee name etc., which can also be displayed along with the salary of that employee. Now we would like to introduce the loop and loop check, as we want to repeat the same module to calculate the salary as many times as required and stop the loop after the salary of all the employees have been calculated. The following variable names are used, in addition to the variable names already defined.

Emp No. : Employee number
Emp Name : Employee name

Step-by-step procedure

Step 1 : Read the values of EmpNo.,Emp Name, BP, HRA, DA,TA, PF.
Step 2 : Calculate net salary using the formula
NS=BP+HRA+DA+TA—PF.
Step 3 : Print Emp No., Emp Name and NS.
Step 4 : Check if there are more employees.
Step 5 : If yes go to step 1. Otherwise go to step 6.
Step 6 : Stop.

Flow chart


START
INPUT EMP NO, EMP NAME, BP, HRA, DA,TA ,PF
NS = BP+HRA+TA+DA-PF
PRINT EMP NO, EMP NAME, NS
IF YOU WANT TO CONTINUE ?
YES
NO
STOP

Example 3 :

Draw a flow chart to find the average of 10 numbers.

In this example we will follow the same procedure as we do in Algebra, ie., first we add all the ten numbers and then we divide the sum by 10. Here we use a loop to add the numbers to get sum. Once the list of numbers is exhausted, then we come out of loop and divide the sum with 10 to get the average.

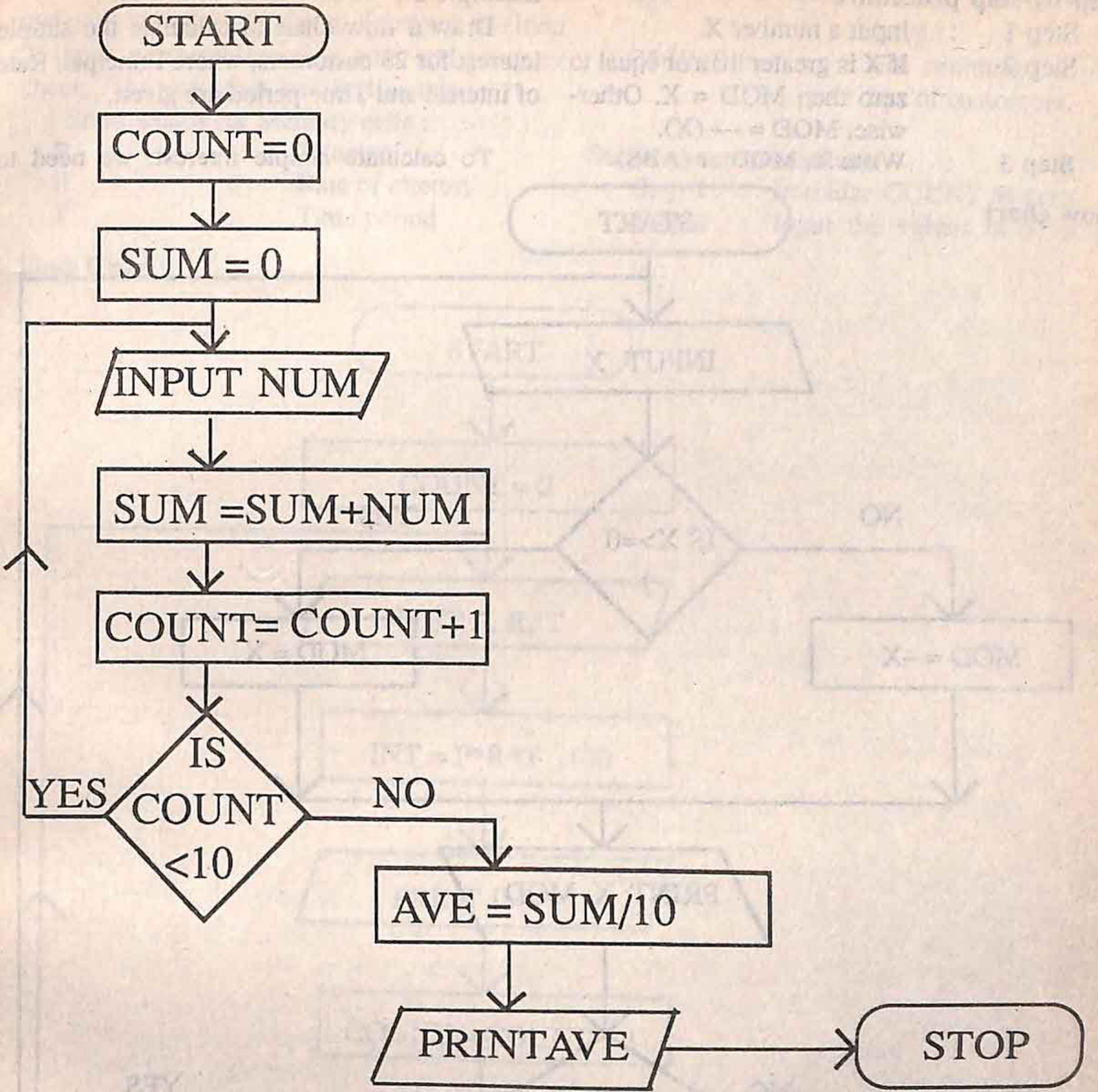
Short names for memory cells :

SUM : Partial Sum
COUNT : Counter for number
NUM : The value of the number
AVE : Average of numbers.

Step-by-step procedure:

Step 1 : Initialize the value of COUNT as zero.
Step 2 : Initialize the value of SUM as zero.
Step 3 : Input the value of NUM.
Step 4 : Add NUM to SUM and call it SUM.
Step 5 : Add 1 to COUNT and call it COUNT.
Step 6 : If COUNT is less than 10 then go to step 3. Otherwise go to step 7.
Step 7 : Divide SUM by 10 to give AVE.
Step 8 : Print AVE.
Step 9 : Stop.

Flow chart



The above flow chart can be divided into three parts : Initialization, Processing and Printing of result. The first two boxes constitute the initialization part. Here the initial values of SUM and COUNT are set. The boxes 4 to 8 constitute the processing part. Here there is one loop consisting of steps 4 to 7.

In this loop, we read the value of each number, calculate the partial sum, keep track of number of numbers processed so far and test to see if all numbers have been processed. In box 8, we calculate the avarage. In box 9, the result is printed.

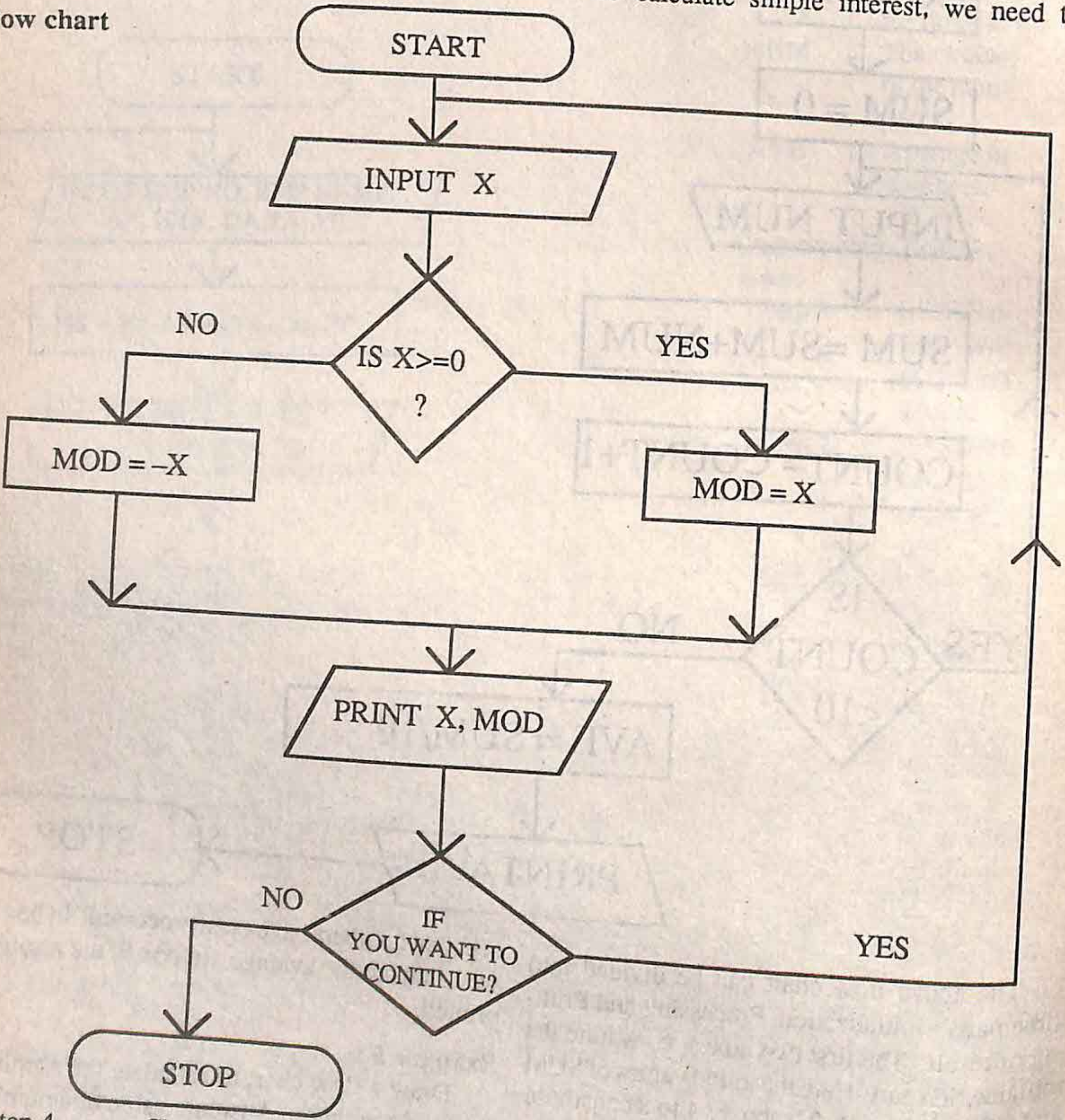
Example 5 :

Draw a flow chart to calculate the absolute value of a number. Absolute value of a number is calculated by checking the number. If the number is greater than or equal to zero then absolute value is same as the number. Otherwise it will be negative of the number.

Step-by-step procedure

Step 1 : Input a number X.
Step 2 : If X is greater than or equal to zero then MOD = X. Otherwise, MOD = — (X).
Step 3 : Write X, MOD or (ABS).

Flow chart



Step 4 : Check if you want to continue further.
Step 5 : If yes, then go to step 1. Otherwise go to step 6.
Step 6 : Stop.

Example 6 :

Draw a flow chart to calculate the simple interest for 25 customers, where Principal, Rate of interest and Time period are given.

To calculate simple interest, we need to know the value of Principal, Rate of interest and Time period. We should have 25 sets of these values because these values will vary from customer to customer. Once we have the necessary information, we will take one set at a time, calculate interest using the formula, I = PRT/100 and print the value of interest. To repeat this loop 25 times, we should have a counter and a counter check. The procedure will be like this :

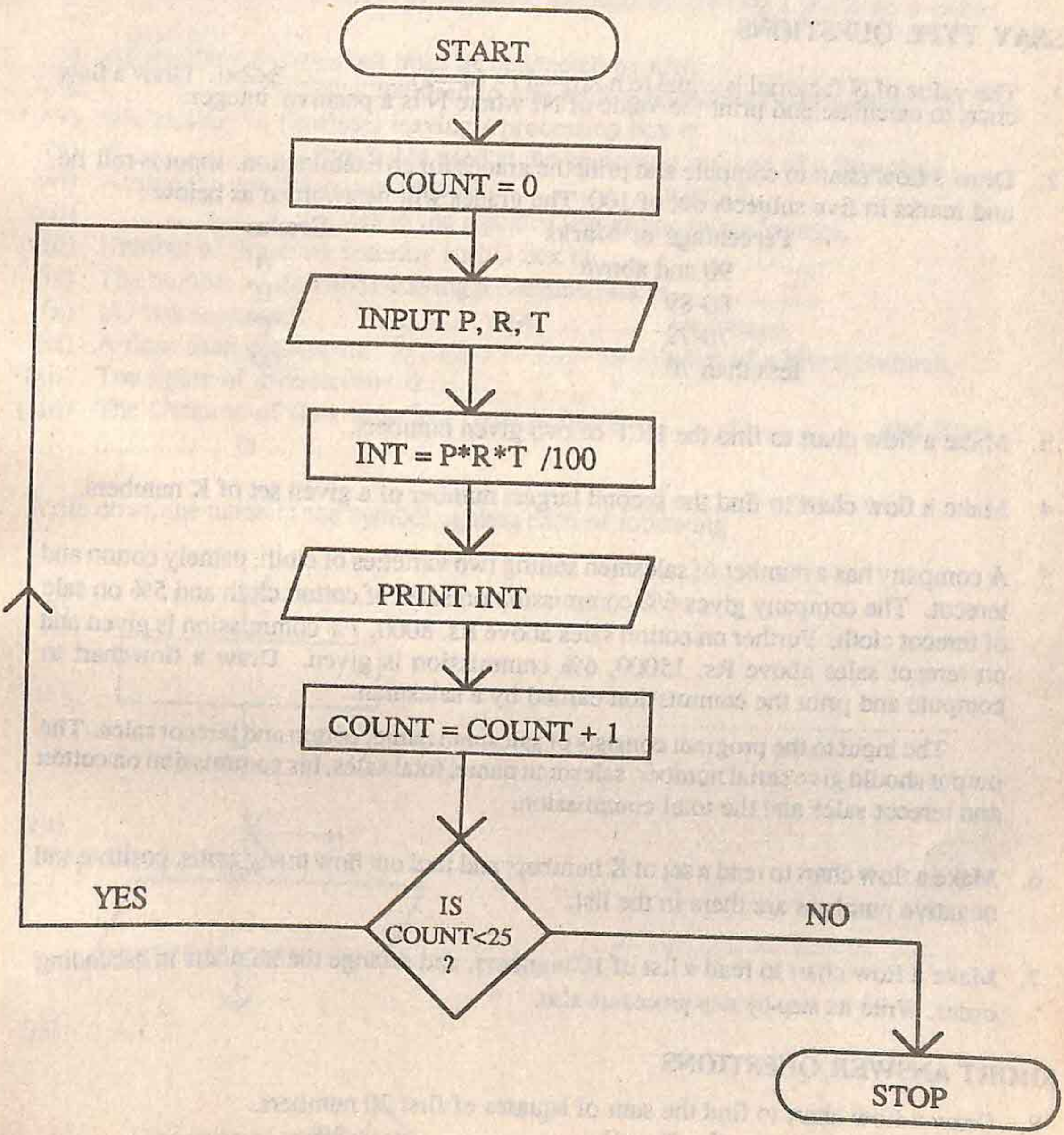
Short names for Memory cells :

P : Principal
R : Rate of interest
T : Time period
INT : Interest to be calculated.
COUNT : For counting the number of customers.

Step-by-step procedure

Step 1 : Initialize COUNT as zero.
Step 2 : Input the values of Principal,

Flow Chart:



Rate of interest and Time period.
Step 3 : Calculate Interest using the formula INT=PRT/100.
Step 4 : Write INT.
Step 5 : Add 1 to COUNT and store the new value in COUNT again.
Step 6 : If the value of COUNT is less than 25 go to step 2. Otherwise go to step 7.
Step 7 : Stop.

EXERCISES

ESSAY TYPE QUESTIONS

1. The value of N factorial is equal to $n \times (n-1) \times (n-2)$..................$3 \times 2 \times 1$. Draw a flow chart to calculate and print the value of N! where N is a positive integer.

2. Draw a flow chart to compute and print the grades for an examination. Input is roll no. and marks in five subjects out of 100. The grades will be awarded as below :

Percentage of Marks	Grades
90 and above	'A'
80-89	'B'
70-79	'C'
less than 70	'D'

3. Make a flow chart to find the HCF of two given numbers.

4. Make a flow chart to find the second largest number of a given set of K numbers.

5. A company has a number of salesmen selling two varieties of cloth, namely cotton and terecot. The company gives 6% commission on sales of cotton cloth and 5% on sale of terecot cloth. Further on cotton sales above Rs. 8000, 7% commission is given and on terecot sales above Rs. 15000, 6% commission is given. Draw a flowchart to compute and print the commission earned by a salesman.

   The input to the program consists of salesman name, cotton and terecot sales. The output should give serial number, salesman name, total sales, his commission on cotton and terecot sales and the total commission.

6. Make a flow chart to read a set of K numbers and find out how many zeros, positive and negative numbers are there in the list.

7. Make a flow chart to read a list of 10 numbers, and arrange the numbers in ascending order. Write its step-by-step procedure also.

SHORT ANSWER QUESTIONS

8. Draw a flow chart to find the sum of squares of first 20 numbers.

   i.e., $1^2 + 2^2 + 3^2 + \dots\dots\dots\dots + 20^2$

9. Draw a flow chart to exchange value of two variables A and B using temporary variables.

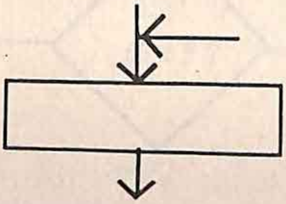
10. Draw a flow chart to print multiplication table of any number N.

OBJECTIVE TYPE QUESTIONS

11. Fill in the blanks :
    - (i) A pictorial representation of sequence of steps for solving a problem is called ....
    - (ii) All symbols in flowchart must be connected by a/an ....
    - (iii) .... are used to join remote portions of a flowchart.
    - (iv) The number of flowlines leaving a processing box is ....
    - (v) A .... symbol is used at the beginning and end of a flowchart.
    - (vi) Arithmetic operations are coded in .... symbol.
    - (vii) .... refers to the repeated use of one or more steps.
    - (viii) Number of flowlines entering an I/O box is ....
    - (ix) The number of flowlines leaving a decision box is.....................
    - (x) I/O box represents .... and .... operations.
    - (xi) A flow chart depicts the .... of solution of a given problem.
    - (xii) The figure of decision box is ....
    - (xiii) The direction of flow in a flow chart is from .... to .... and from .... to ....

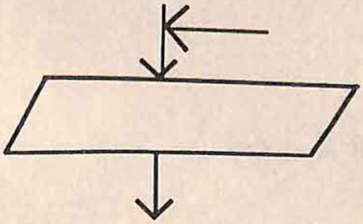
12. Write down the name of the symbol against each of following :

(i)



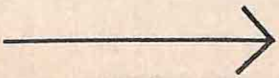
....

(ii)



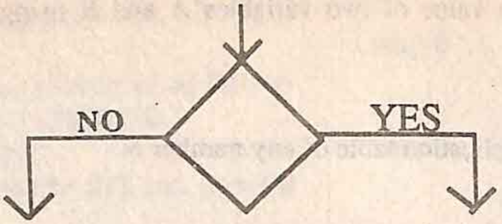
....

(iii)



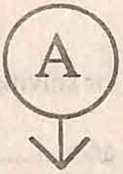
....

(iv)



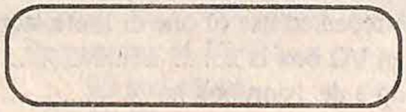
....

(v)



....

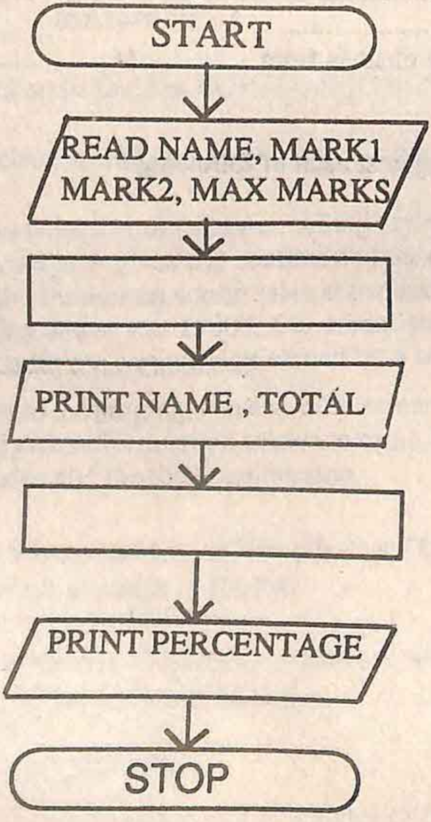
(vi)



....

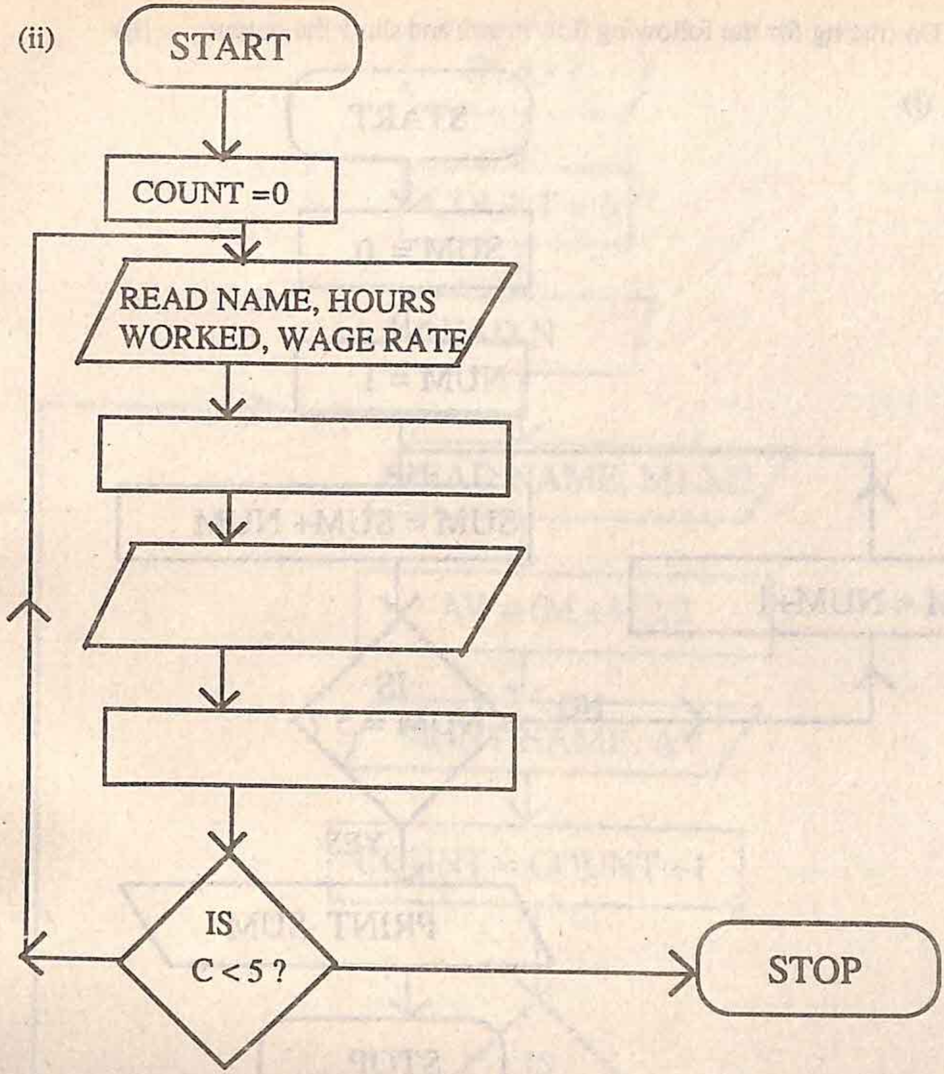
13. Complete the following flow charts and write in one line the purpose of flow charts.

(i)



PURPOSE : ....

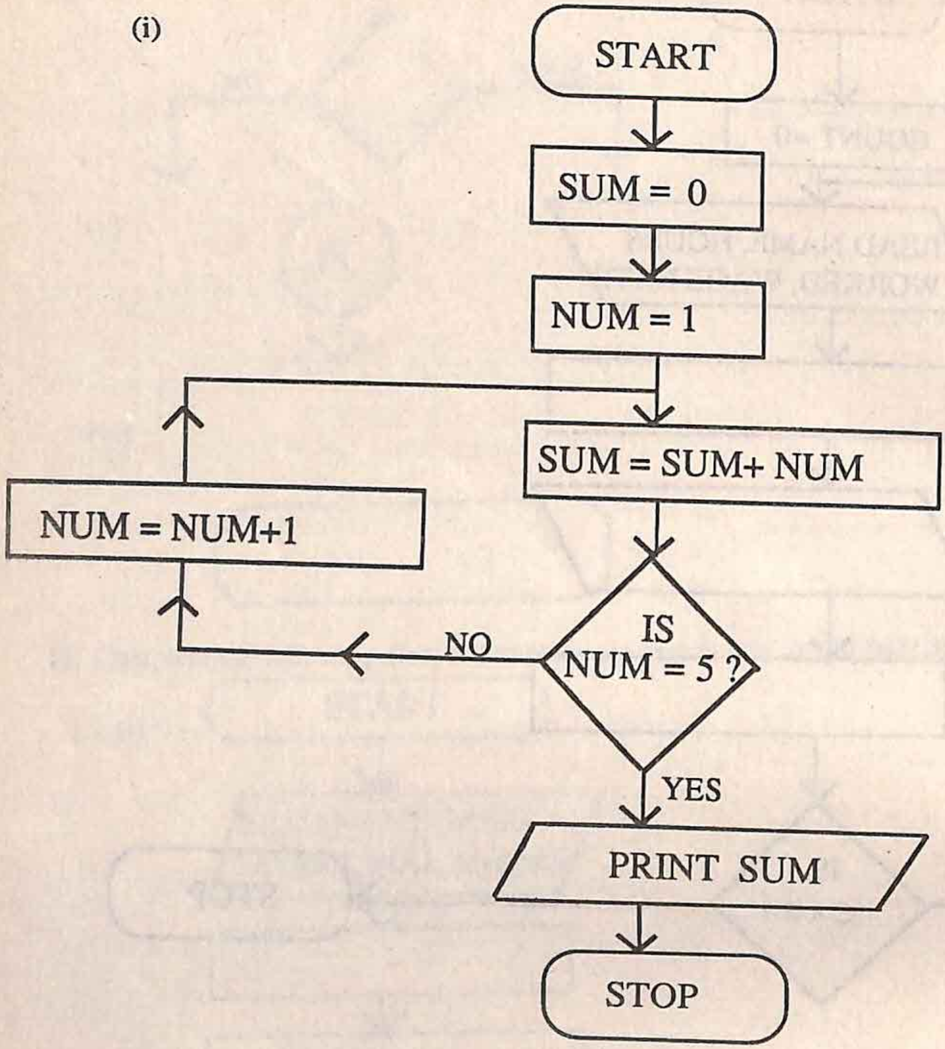
(ii)



PURPOSE : ....

14. Do tracing for the following flow charts and show the output.

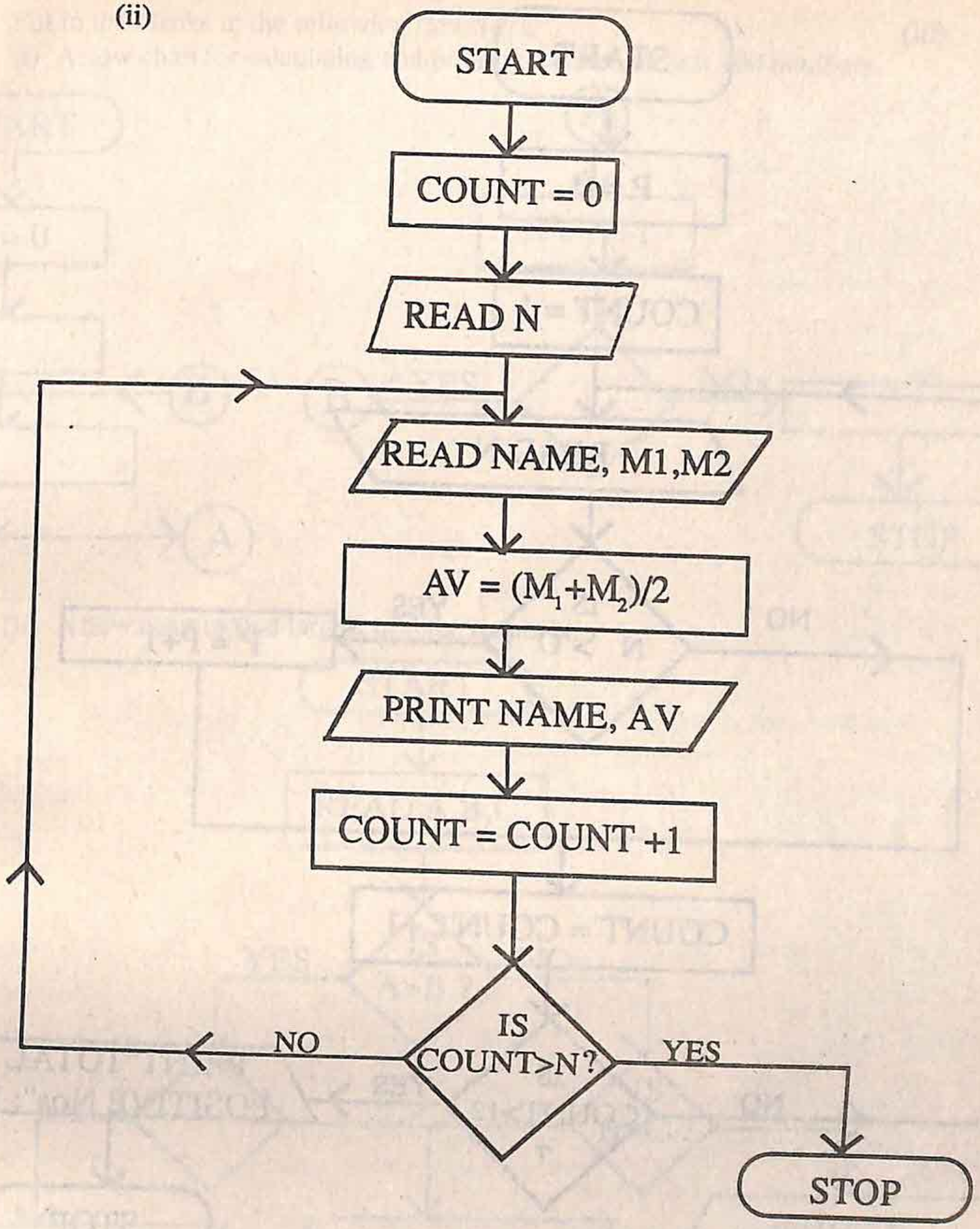
(i)



Also trace the above flow chart with following conditions :-

(i) NUM < 5
(ii) NUM > 5
(iii) NUM >= 5

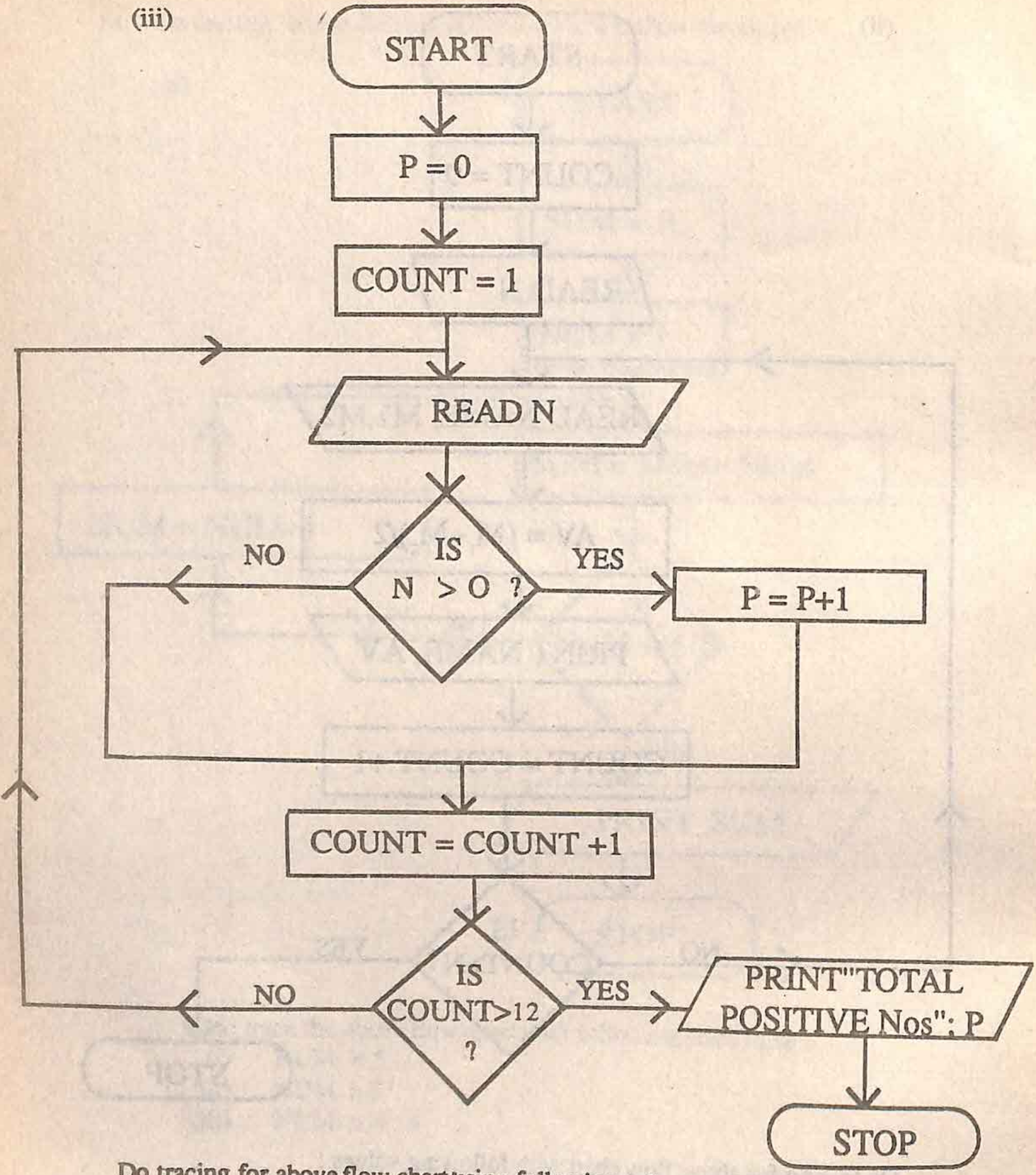
(ii)



Do tracing for above flow chart with following values :

N = 3
AMIT, 20,40
AJAY, 15,17
ANU, 25,30

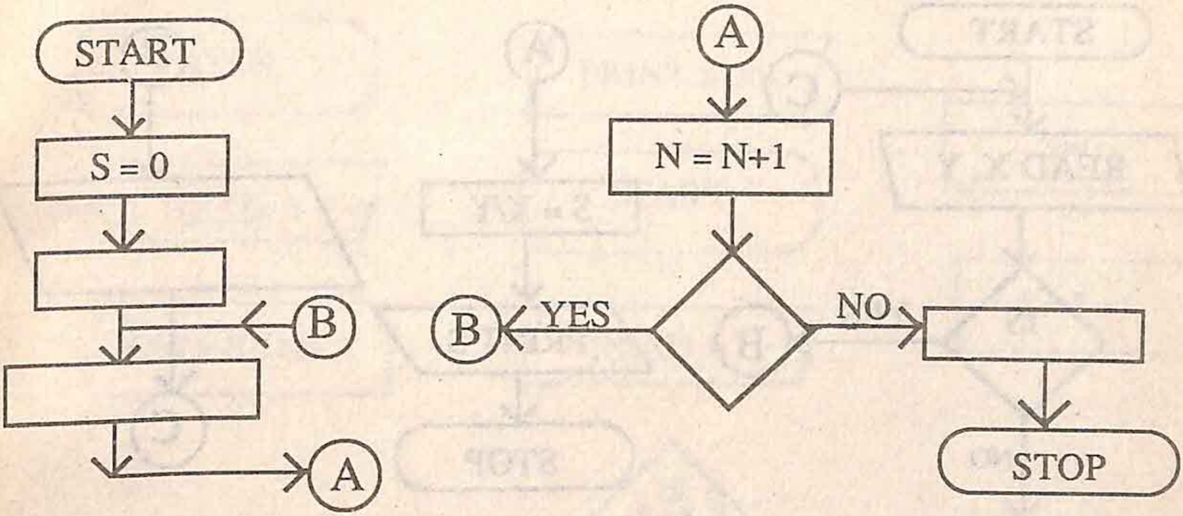
(iii)



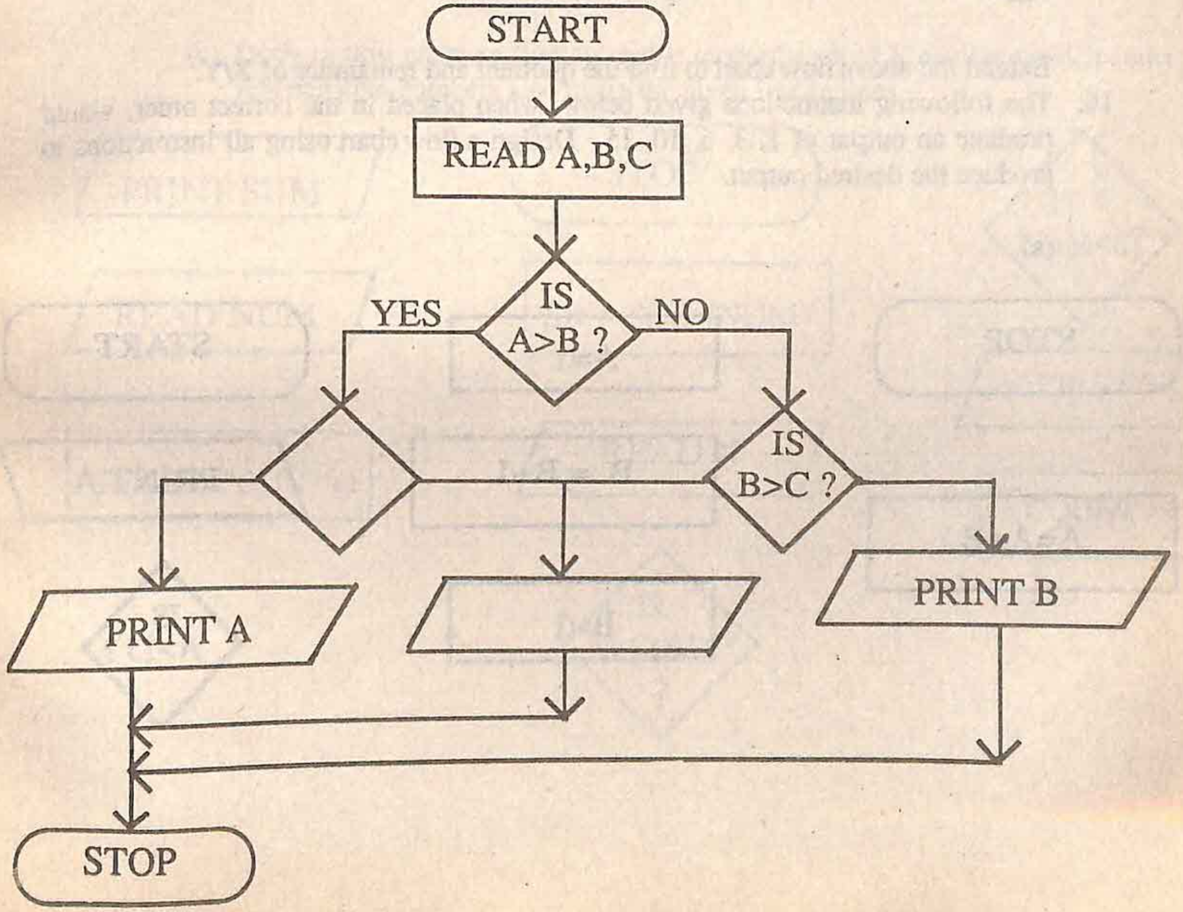
Do tracing for above flow chart using following values and find the output :
5, 0, — 62, 29, 54, 79, 48, — 9,— 8, 0, 59, 62

15. Fill in the blanks in the following flowcharts :

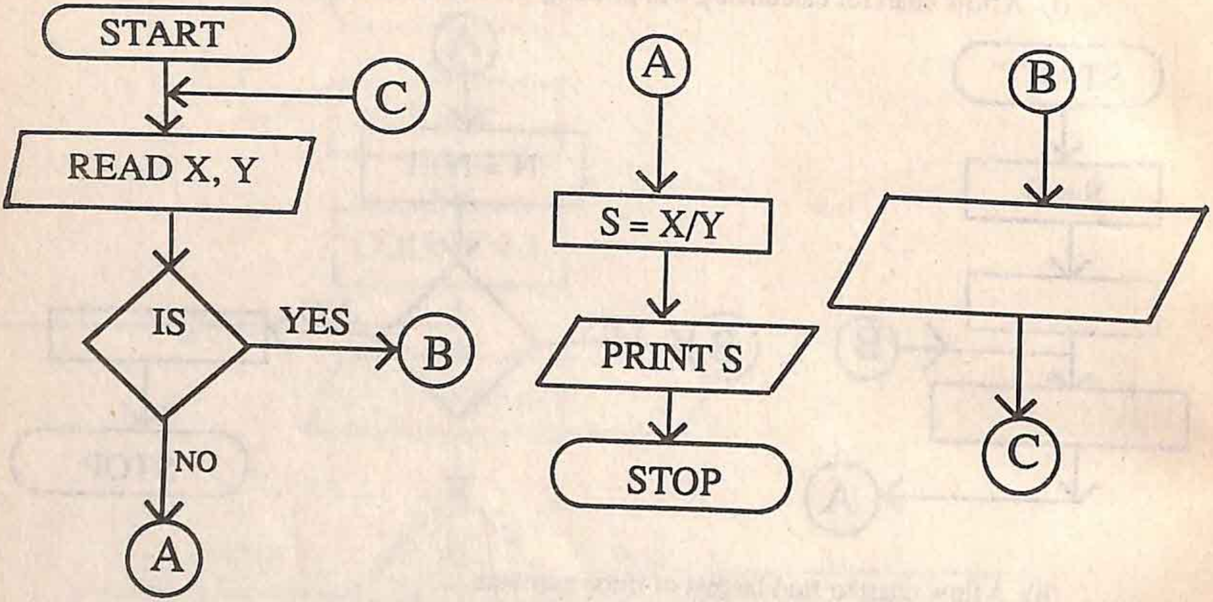
(i) A flow chart for calculating and printing the sum of first 100 numbers.



(ii) A flow chart to find largest of three numbers.



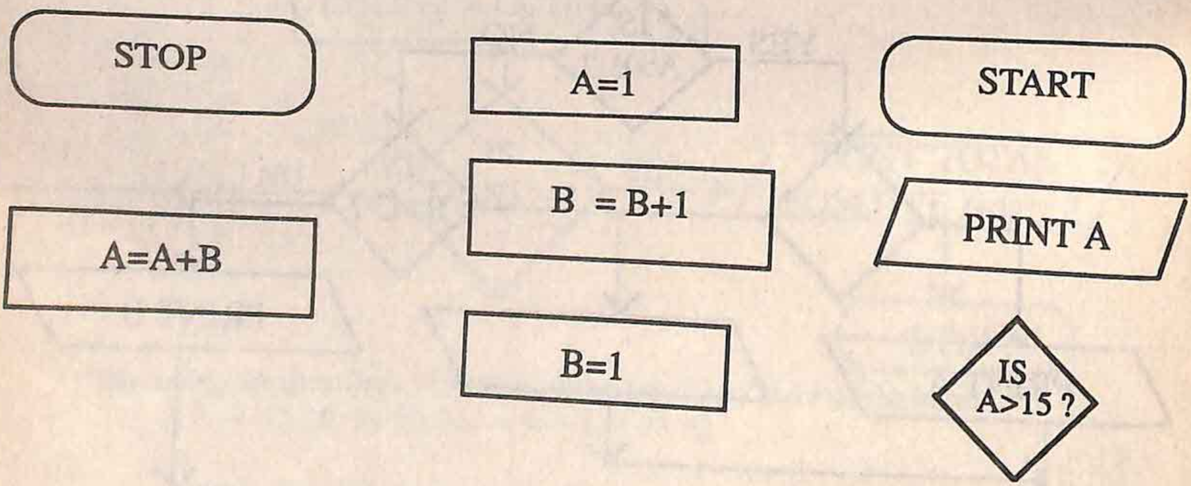
(iii) A flowchart to find the division of X by Y is given below Give proper check whether Y is equal to zero or not.



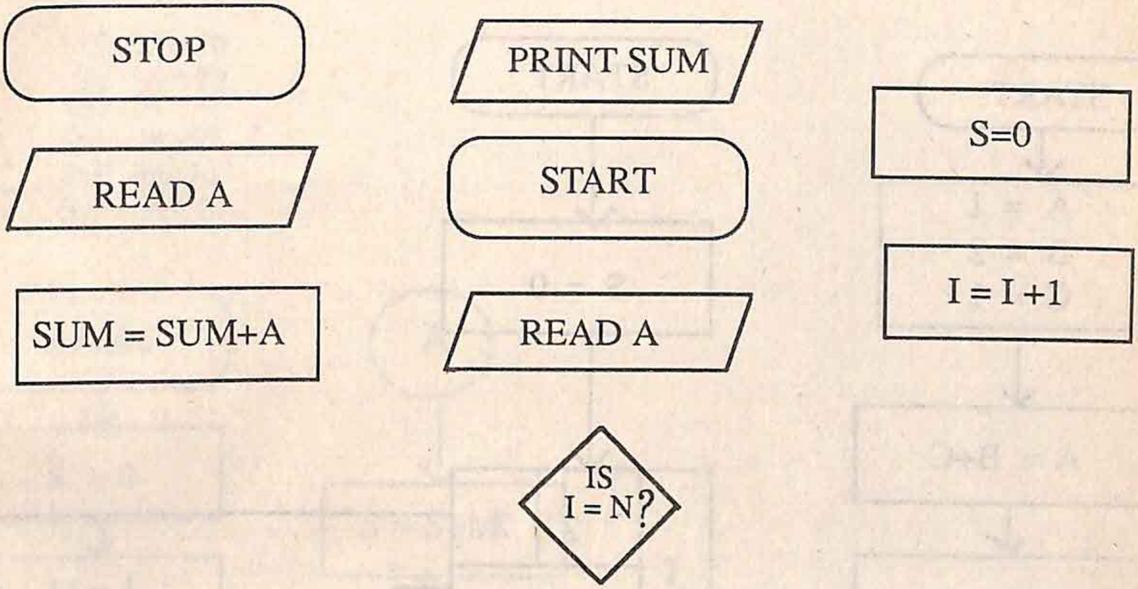
Extend the above flow chart to find the quotient and remainder of X/Y.

16. The following instructions given below, when placed in the correct order, would produce an output of 1, 3, 6, 10, 15. Design a flow chart using all instructions to produce the desired output.

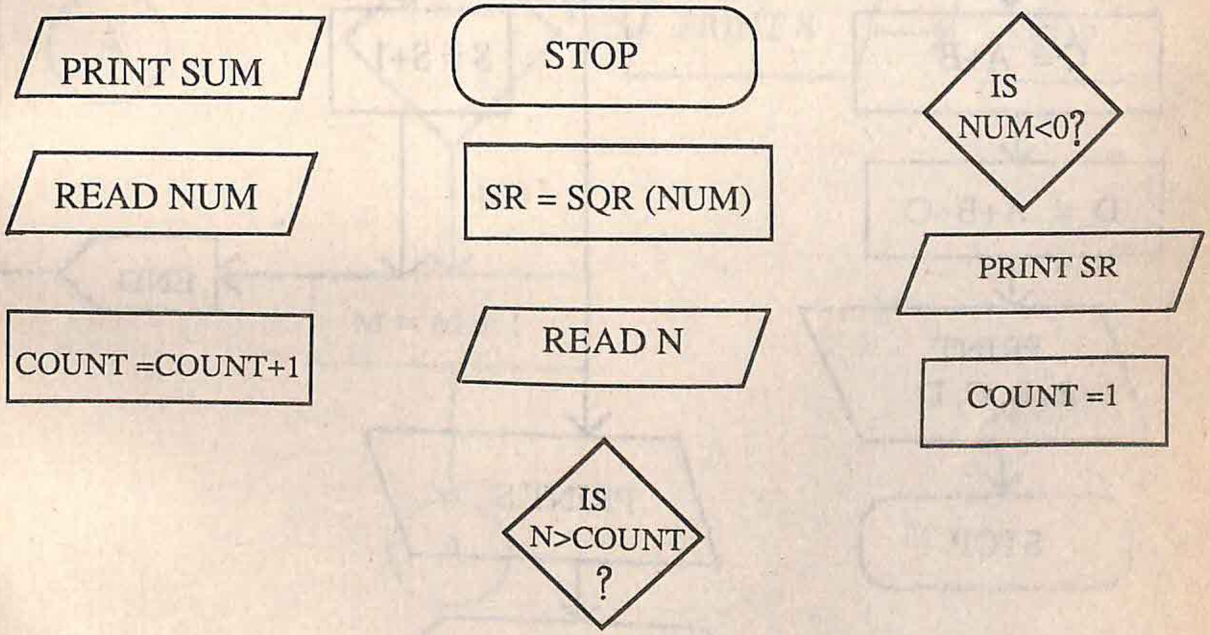
(a)



(b) Design the flow chart by placing the following instructions in correct order to produce the sum of first N natural numbers.

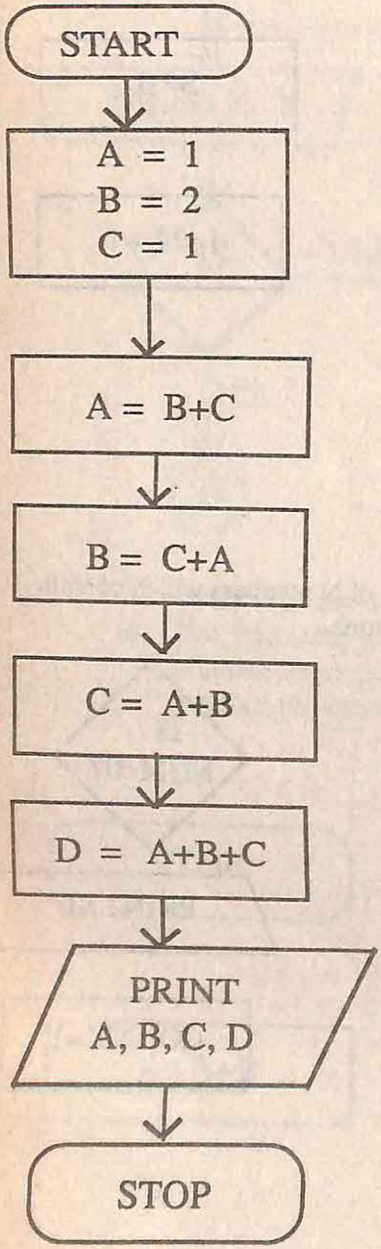


(c) Design a flow chart for finding square roots of a set of N numbers which contain few negative numbers also using the given instructions.

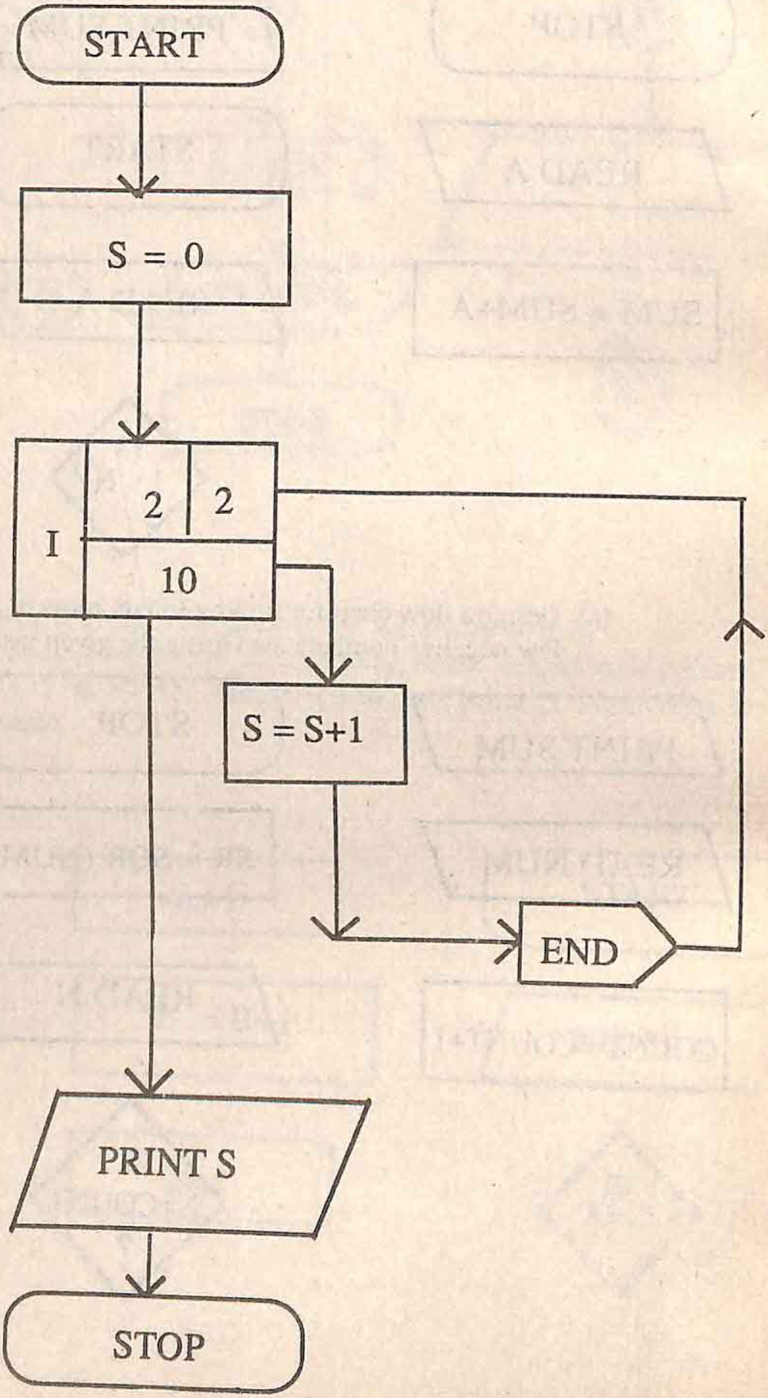


17. Give the outputs of the following flowcharts.

(i)



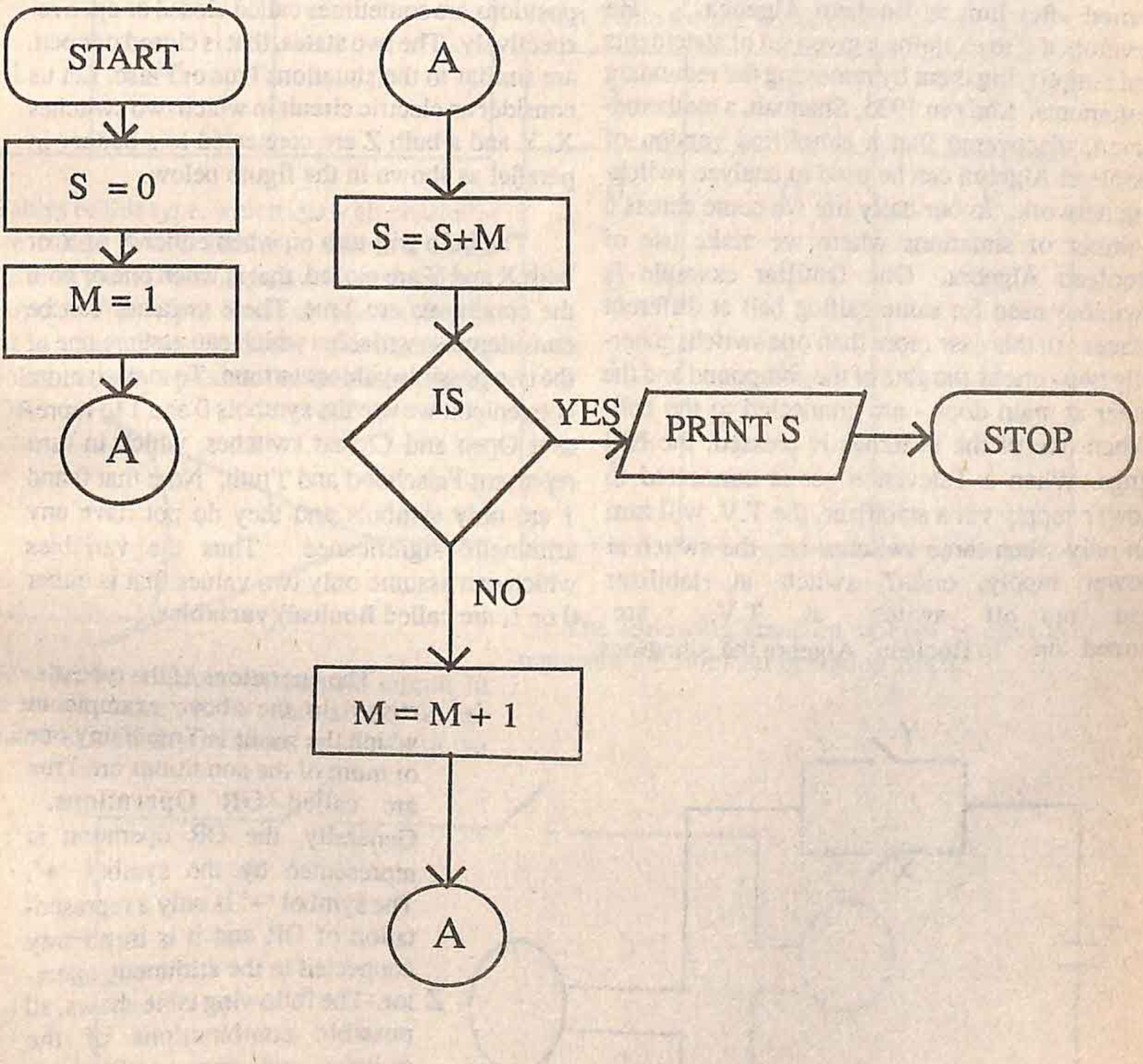
(ii)



18. Following is the flow chart to calculate the sum of series 1 + 2 + 3 +.........n

Write what will be the output if you use the following condition in the decision box :

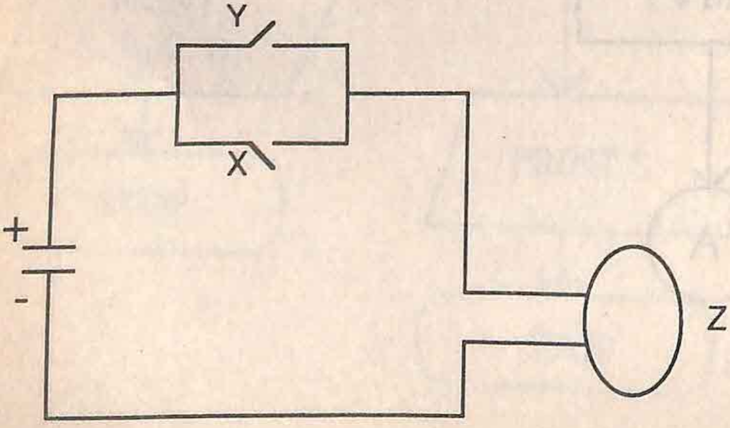
(i) $M \geq 10$
(ii) $M > 20$
(iii) $M = 0$
(iv) $M \geq 20$
(v) $M \leq 20$
(vi) $M < 20$



COMPUTER LOGIC

In this chapter we will introduce an Algebra which is widely used in designing electronic circuits in computers. This Algebra was developed by George Boole (1815-1864), and was named after him as **Boolean Algebra.** He developed it to examine a given set of statements and simplifying them by removing the redundant statements. Later in 1935, Shannan, a mathematician, discovered that a simplified version of Boolean Algebra can be used to analyze switching network. In our daily life we come across a number of situations where we make use of Boolean Algebra. One familiar example is switches used for same calling bell at different places. In this case more than one switch, generally two - one at the gate of the compound and the other at main door - are connected to the bell. When one of the switches is pressed, the bell rings. When a Television set is connected to power supply via a stabilizer, the T.V. will turn on only when three switches i.e., the switch at power supply, on/off switch at stabilizer and on/ off switch at T.V., are turned on In Boolean Algebra the situations like the former example are called OR situations and the latter are called AND situations. An electric switch can be in only one of the two positions, either on or off and these two positions are sometimes called closed or open respectively. The two states, that is closed or open, are similar to the situations True or False. Let us consider an electric circuit in which two switches X, Y and a bulb Z are connected to a battery in parallel as shown in the figure below.

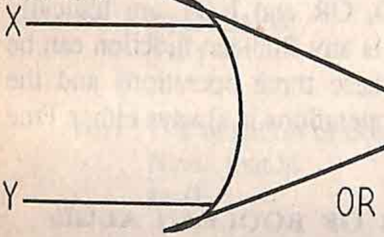
The bulb will turn on when either X or Y or both X and Y are closed, that is when one or both the conditions are True. These switches can be considered as variables which can assume one of the two possible values at a time. To make it more convenient, we use the symbols 0 and 1 to represent Open and Closed switches, which in turn represent Falsehood and Truth. Note that 0 and 1 are only symbols and they do not have any arithmetic significance. Thus the variables which can assume only two values that is either 0 or 1, are called **Boolean variables.**



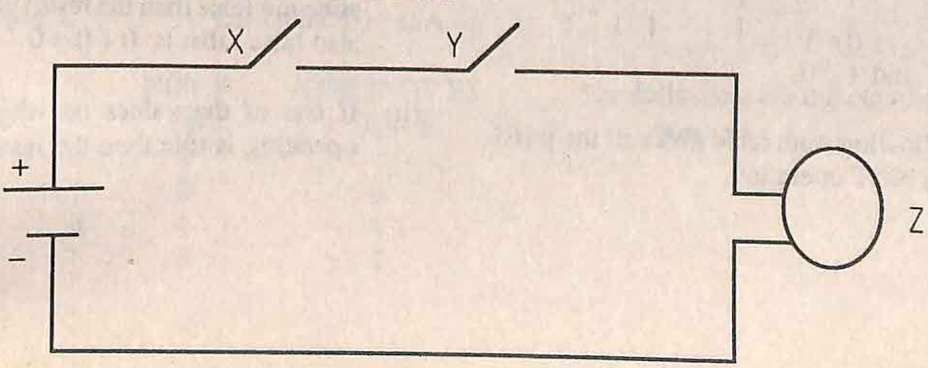
The operations of the type discussed in the above example in which the result is True if any one or more of the conditions are True are called **OR Operations.** Generally, the OR operation is represented by the symbol '+'. The symbol '+' is only a representation of OR and it is in no way connected to the arithmetic operator. The following table shows, all possible combinations of the switches and corresponding output or result. Note that 0 represents a switch which is open and 1 represents a switch which is closed and 0 for bulbrepresents off and 1 represents on (last column).

X	Y	Z = X + Y
0	0	0
0	1	1
1	0	1
1	1	1

Tables of this type, which show all possibilities are called **Truth Tables,** that is a Truth table shows the set of all inputs with corresponding outputs. The output is also always either 0 or 1, that is either False or True. The following symbol is used to represent the Boolean Operation OR.



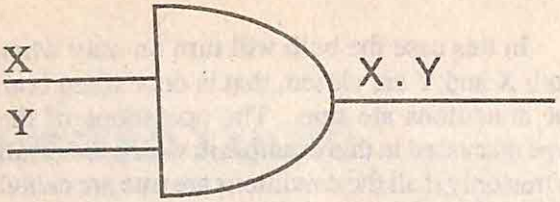
Now let us consider an electric circuit in which two switches X, Y and a bulb Z are connected to a battery in series as shown in the figure below.



In this case the bulb will turn on only when both X and Y are closed, that is only when both the conditions are true. The operations of the type discussed in this example in which the result is true only if all the conditions are true are called **AND Operations.** The AND operation is represented by '.' . The '.' is similar to the multiplication sign of arithmetic. The only difference is, in Boolean Algebra the operands are always either zero or one. The following is the Truth Table for the series circuit given below :

X	Y	Z=X.Y
0	0	0
0	1	0
1	0	0
1	1	1

The following standard symbol is used to represent the Boolean operation AND.



The operations AND and OR require at least two variables and hence these are called **Binary Operators.** If the number of variables is more than two, one has to use these operations repeatedly to get the result. Now let us discuss another important operation of Boolean Algebra called NOT. We know that the capital of India is Delhi. So the proposition 'Delhi is capital of India' is true and the proposition 'Delhi is not the capital of India' is false. Then, the proposition 'Delhi is not the capital of India' is called the negated proposition of 'Delhi is the capital of India' and vice-versa. The same type of situations come into picture in electrical circuits also. There are some electrical circuits in which switches have normally closed contacts. These contacts open when the circuit is in operation. So the normally closed contact represents a negation of the normally open contact of the same circuit. This is called a **NOT Operation.** If a Boolean Variable A=0 then NOT A=1.

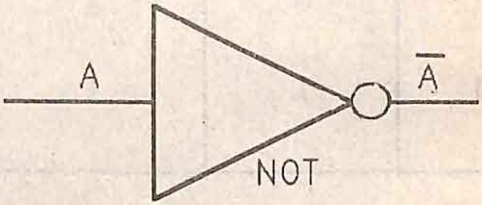
This, i.e.,NOT A, is usually represented as $\overline{A}$. The negation, which is also called **complementation,** converts the symbol 0 to 1 and 1 to 0. That is

$$\overline{0} = 1$$
$$\text{and } \overline{1} = 0$$

The following truth table gives all the possibilities of a NOT operation.

A	NOT A = $\overline{A}$
0	1
1	0

The following standard symbol is used to represent the Boolean operation NOT.



We saw that the operations AND and OR are binary operations as they require two variables to operate upon. But the NOT operation requires only a single variable to operate upon. Hence NOT is called **Unary Operator.** The logical operations AND, OR and NOT are logically complete. That is any Boolean function can be realized using these three operations and the output of these operations is always either True or False.

POSTULATES OF BOOLEAN ALGEBRA

(i) If both the values on which OR is operating are false then the result or output is also false, that is $0 + 0 = 0$.

(ii) If one of the values on which OR is operating is true then the result is true, that is $0 + 1 = 1$ or $1 + 0 = 1$.

(iii) If both the values on which OR is operating are true then the result is also true, that is
$1 + 1 = 1$.

(iv) If both the values on which AND is operating are false then the result is also false, that is
$0.0 = 0$.

(v) If one of the values on which AND is operating is false then the result is false, that is
$0.1 = 0$
$1.0 = 0$.

(vi) If both the values on which AND is operating are true then the result is also true, that is
$1.1 = 1$.

(vii) The negation or complement of false is true, that is
$\overline{0} = 1$.

(viii) The negation or complement of true is false, that is
$\overline{1} = 0$.

A few more examples of Truth Tables are discussed below.

Example 1. A OR (NOT B) i.e., $A + \overline{B}$

A	B	NOT B ($\overline{B}$)	A OR (NOT B) $A + \overline{B}$
0	0	1	1
0	1	0	0
1	0	1	1
1	1	0	1

Example 2 NOT (A AND B) i.e., $\overline{(A.B)}$

A	B	A AND B (A.B)	NOT (A AND B) $\overline{(A.B)}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Example 3 NOT (A OR B) i.e., $\overline{(A \text{ OR } B)}$

A	B	A OR B (A + B)	NOT (A OR B) $\overline{(A \text{ OR } B)}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

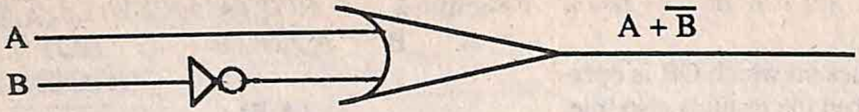
Example 4

(A OR B) AND (A OR C) i.e., $(A + B).(A + C)$

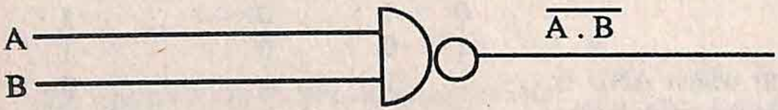
A	B	C	A OR B A + B	A OR C A + C	(A OR B) AND (A OR C) (A + B). (A + C)
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	1	1	1
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	1	1	1
1	1	1	1	1	1

The following are the circuits for the above example :

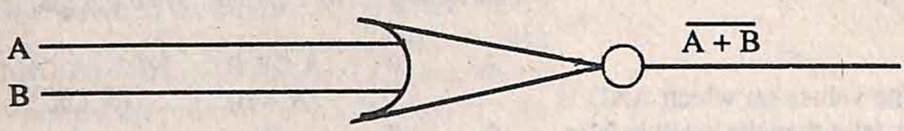
1.



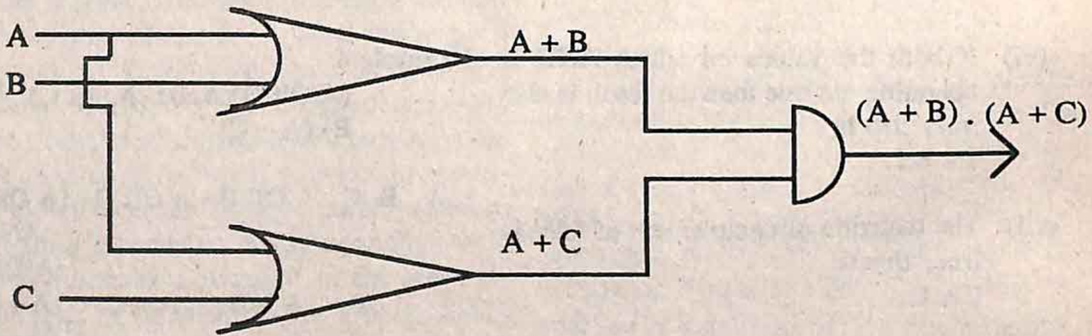
2.



3.



4.



For any Boolean Variable A, B or C the following are true. The proofs are immediate from the postulates and also from Truth Tables.

1. Commutative Law

(a) A+B = B+A

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

A	B	B+A
0	0	0
0	1	1
1	0	1
1	1	1

(b) A.B = B.A

A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

A	B	B.A
0	0	0
0	1	0
1	0	0
1	1	1

(2) **Distributive Laws**

(a) A. (B + C) = A.B + A.C

A	B	C	B+C	A.(B+C)
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	0	0
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

A	B	C	A.B	A.C	(A.B)+ (A.C)
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	0	1	1
1	1	0	1	0	1
1	1	1	1	1	1

3 A + 0 = A

A	0	A+0
0	0	0
1	0	1

4 A.1 = A

A	1	A.1
0	1	0
1	1	1

5 $A + \overline{A} = 1$

A	$\overline{A}$	$A+\overline{A}$
0	1	1
1	0	1

6 $A . \overline{A} = 0$

A	$\overline{A}$	$A.\overline{A}$
0	1	0
1	0	0

Now let us prove some of the important theorems of Boolean Algebra.

Theorem 1a : A + A = A
R.H.S. = A
= A + 0 (From 3)
= $A + (A.\overline{A})$ (From 6)
= $(A + A).(A + \overline{A})$ (From 2b)
= (A + A).1 (From 5)
= A + A (From 4)
= L.H.S.

Theorem 1b : A.A = A
R.H.S. = A
= A.1 (From 4)
= $A. (A + \overline{A})$ (From 5)
= $(A.A) + (A . \overline{A})$ (From 2a)
= A.A + 0 (From 6)
= A.A
= L.H.S.

Theorem 2a : A + 1 = 1
L.H.S. = A + 1
= $A + (A + \overline{A})$
= $(A + A) + \overline{A}$
= $A + \overline{A}$
= 1
= R.H.S.

Theorem 2b : A.0 = 0
L.H.S. = A.0
= (A.0) + 0
= $(A.0) + (A.\overline{A})$
= $A.(0 + \overline{A})$
= $A. \overline{A}$
= 0
= R.H.S.

Theorem 3a : A + A.B = A
L.H.S. = A + A.B
= A.1 + A.B
= A.(1 + B)
= A.1
= A
= R.H.S.

Theorem 3b : $A.(A+B)=A$

$$\begin{aligned} \text{L.H.S.} &= A.(A+B) \\ &= (A+0).(A+B) \\ &= A+(0.B) \\ &= A+0 \\ &= A \\ &= \text{R.H.S.} \end{aligned}$$

Theorem 4 : $\bar{\bar{A}} = A$

Proof When $A=0$, $\overline{A}=1$ and $\bar{\bar{A}}=\overline{1}=0=A$
When $A=1$, $\overline{A}=0$ and $\bar{\bar{A}}=\overline{0}=1=A$

Theorem 5 : $A+(\overline{A}.B)=A+B$

$$\begin{aligned} \text{L.H.S.} &= A+(\overline{A}.B) \\ &= (A+\overline{A}).(A+B) \\ &= 1.(A+B) \\ &= A+B \\ &= \text{R.H.S.} \end{aligned}$$

Theorem 6 : $A.(\overline{A}+B)=A.B$

$$\begin{aligned} \text{L.H.S.} &= A.(\overline{A}+B) \\ &= (A.\overline{A})+(A.B) \\ &= 0+(A.B) \\ &= A.B \\ &= \text{R.H.S.} \end{aligned}$$

De Morgan's Laws : These De Morgan's Laws are similar to the De Morgan Laws in set theory, i.e.,

$$\overline{(A\cup B)}=\overline{A}\,.\,\overline{B}$$

and $$\overline{(A\cap B)}=\overline{A}+\overline{B}$$

These theorems are used to simplify expressions so that final output which is either zero or one, can be determined easily.

Theorem 1 : The complement of the sum (OR) of two variables is equal to the product (AND) of the complement of the variables, i.e,

$$\overline{X+Y}=\overline{X}.\overline{Y}$$

Proof : We know that

$$X.\overline{X} = 0$$

and $$X+\overline{X} = 1$$

So, if we show

$$(X+Y).(\overline{X}.\overline{Y})=0$$

and $$(X+Y)+(\overline{X}.\overline{Y})=1$$

then we can show that $(\overline{X+Y})=\overline{X}.\overline{Y}$

$$\begin{aligned} (X+Y).(\overline{X}.\overline{Y}) &= \left[(X.\overline{X})+(Y.\overline{X})\right].\,\overline{Y} \\ &= Y.\overline{X}.\overline{Y} \qquad \text{(Distributive Law)} \\ &= (\overline{X}.Y).\,\overline{Y} \\ &= \overline{X}.(Y.\overline{Y}) \\ &= \overline{X}.0=0 \end{aligned}$$

Now let us consider

$$\begin{aligned} &(X+Y)+(\overline{X}.\overline{Y}) \\ &=(X+Y+\overline{X}).(X+Y+\overline{Y}) \\ &=(1+Y).(\overline{X}+1) \\ &=1.1=1 \end{aligned}$$

As the two relations are proved hence the main theorem is proved.

Theorem 2 : The complement of the product (AND) of the two variables is equal to the sum (OR) of the complements of the variables. That is

$$\overline{(X.Y)}=\overline{X}+\overline{Y}$$

Proof : To prove this theorem we have to show that

$$(X.Y).(\overline{X}+\overline{Y})=0$$

and $$(X.Y)+(\overline{X}+\overline{Y})=1$$

Now, $(X.Y).(\overline{X}+\overline{Y})$

$$\begin{aligned} &=(X.Y).\,\overline{X}+(X.Y).\,\overline{Y} \\ &=Y.X\,\overline{X}+X.Y.\overline{Y} \\ &=Y.(X.\overline{X})+X.(Y.\overline{Y}) \\ &=Y.0+X.0 \\ &=0 \end{aligned}$$

Also $(X.Y)+(\overline{X}+\overline{Y})$

$$=(X+(\overline{X}+\overline{Y})).\,(Y+(\overline{X}+\overline{Y}))$$

$=(X+\overline{X}+\overline{Y}).(Y+\overline{Y}+\overline{X})$

$=(1+\overline{Y}).(1+\overline{X})$

($\because X+\overline{X}=1$ and, $Y+\overline{Y}=1$)

$= 1.1$

$= 1$

Which proves the theorem.

In principle all the theorems discussed above can be proved through the truth tables by considering all possible cases. This method of proving theorems through truth table is called method of exhaustive enumeration. Now let us verify the DeMorgan's laws with the help of Truth Tables.

(i) $\overline{X+Y}=\overline{X}.\overline{Y}$

X	Y	X + Y	$\overline{X}$	$\overline{Y}$	$\overline{X+Y}$	$\overline{X}.\overline{Y}$
0	0	0	1	1	1	1
0	1	1	1	0	0	0
1	0	1	0	1	0	0
1	1	1	0	0	0	0

(ii) $\overline{X.Y}=\overline{X}+\overline{Y}$

X	Y	X.Y	$\overline{X}$	$\overline{Y}$	$\overline{X.Y}$	$\overline{X}+\overline{Y}$
0	0	0	1	1	1	1
0	1	0	1	0	1	1
1	0	0	0	1	1	1
1	1	1	0	0	0	0

The theorem obtained by interchanging '.', '+', 1 and 0 in a theorem is called the dual. For example, the dual of $X.(Y+Z) = (X.Y) + (X.Z)$ is $X + (Y.Z) = (X+Y).(X+Z)$. The following table shows some of the theorems and their duals. Observe that if you interchange '.', '+', 1 and 0 in the theorems of left hand side, you get the theorems of the right hand side and vice versa.

$$= 1.(X+Y)$$
$$= X + Y$$

Example 2 : $X.(\overline{X}+Y) = X.Y$

$$X.(\overline{X}+Y) = (X.\overline{X}) + (X.Y)$$
$$= 0 + (X.Y) = X.Y$$

Example 3 : $(X+Y).(X+Z) = X + Y.Z$

Theorem	Dual Theorem
1. $A + 0 = A$	1. $A.1 = A$
2. $A + B = B + A$	2. $A.B = B.A$
3. $A.(B+C) = (A.B) + (A.C)$	3. $A + (B.C) = (A+B).(A+C)$
4. $A + A = 1$	4. $A.A = 0$
5. $A + A = A$	5. $A.A = A$
6. $A + 1 = 1$	6. $A.0 = 0$
7. $A + A.B = A$	7. $A.(A+B) = A$
8. $A + \overline{A}.B = A + B$	8. $A.(\overline{A}+B) = A.B$
9. $\overline{A+B} = \overline{A}.\overline{B}$	9. $\overline{A.B} = \overline{A}+\overline{B}$
10. $A + (B+C) = (A+B) + C$	10. $A.(B.C) = (A.B).C$

Now let us see how various postulates and theorems discussed above can be used to simplify the Boolean expressions.

Example 1: $X + (\overline{X}.Y) = X + Y$ where X and Y are boolean variables.

$$X + (\overline{X}.Y) = (X+\overline{X}).(X+Y)$$

(Distributive Law)

Where X, Y and Z are boolean variables.

Solution :

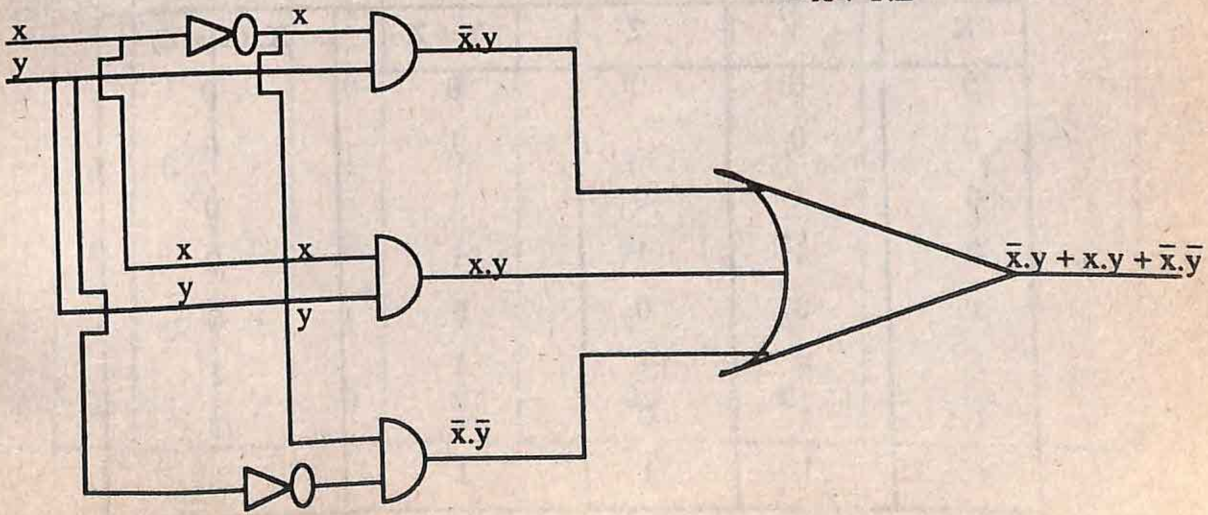
$$(X+Y).(X+Z) = X.(X+Z) + Y.(X+Z)$$

(Distributive Law)

$$= X.X + X.Z + Y.X + Y.Z$$

(Distributive Law)

$$= X.(1+Z) + Y.X + Y.Z$$
$$= X + Y.X + Y.Z$$
$$= X.(1+Y) + Y.Z$$
$$= X + Y.Z$$



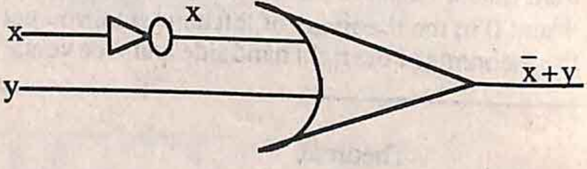
Example 4 : Simplify the following Boolean expression

$$\overline{X}.Y + X.Y + \overline{X}.\overline{Y}$$

Solution :

$$\overline{X}.Y + X.Y + \overline{X}.\overline{Y}$$
$$= (X + X).Y + \overline{X}.\overline{Y}$$
$$= 1.Y + \overline{X}.\overline{Y}$$
$$= Y + \overline{X}.\overline{Y} = Y + \overline{Y}.\overline{X}$$
$$= Y + \overline{X} \because X + \overline{X}.\overline{Y} = X + \overline{Y}$$

If we want to show the above expression diagrammatically we must use three AND gates, two NOT gates and OR gate as shown above.



After simplification, we saw that the same result can be obtained by using the expression $Y + \overline{X}$, using only one NOT gate and one OR gate. This simplifies the circuitary of the computer.

Example 5 : Prepare truth table for the following Boolean expression.

(i) $\overline{X}.Y + X.\overline{Y}$

X	$\overline{X}$	Y	$\overline{Y}$	$\overline{X}.Y$	$X.\overline{Y}$	$\overline{X}.Y + X.\overline{Y}$
1	0	0	1	0	1	1
1	0	1	0	0	0	0
0	1	1	0	1	0	1
0	1	0	1	0	0	0

(ii) X. (Y + Z)

X	Y	Z	Y + Z	X. (Y + Z)
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	0	0
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

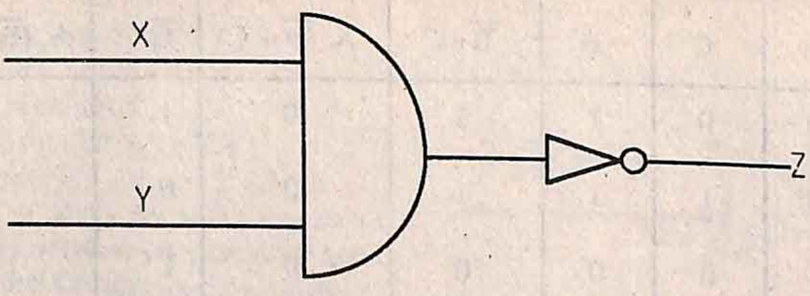
(iii) A. $(\overline{B} + C) + \overline{C}$

A	B	C	$\overline{B}$	$\overline{B} + C$	A. $(\overline{B} + C)$	$\overline{C}$	A. $(\overline{B} + C) + \overline{C}$
0	0	0	1	1	0	1	1
0	0	1	1	1	0	0	0
0	1	0	0	0	0	1	1
0	1	1	0	1	0	0	0
1	0	0	1	1	1	1	1
1	0	1	1	1	1	0	1
1	1	0	0	0	0	1	1
1	1	1	0	1	1	0	1

(iv) X. $(\overline{Y} + \overline{Z}) + X.\overline{Y}$

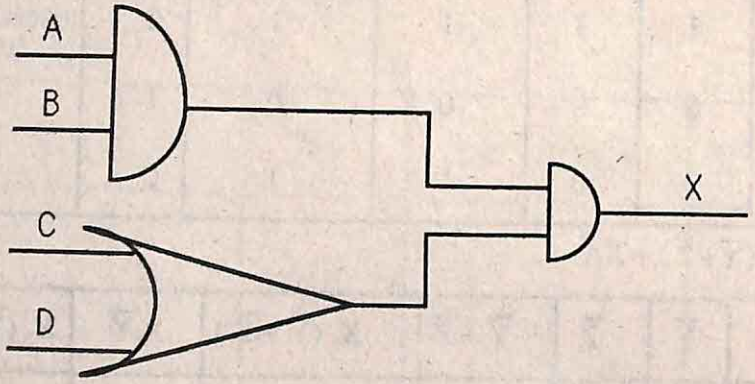
X	Y	Z	$\overline{Y}$	$\overline{Z}$	$\overline{Y} + \overline{Z}$	X. $(\overline{Y} + \overline{Z})$	X.$\overline{Y}$	X. $(\overline{Y} + \overline{Z}) + X.\overline{Y}$
0	0	0	1	1	1	0	0	0
0	0	1	1	0	1	0	0	0
0	1	0	0	1	1	0	0	0
0	1	1	0	0	0	0	0	0
1	0	0	1	1	1	1	1	1
1	0	1	1	0	1	1	1	1
1	1	0	0	1	1	1	0	1
1	1	1	0	0	0	0	0	0

Also if we know the logic circuit we can get the Boolean expression. For example, the Boolean expression of



the logic circuit is $Z=\overline{X.Y}$

Example 6 : Obtain the Boolean expression for the following logic circuit.



Solution : The output of the first AND gate is A.B and the output of the OR gate is C+D. Now A.B and C + D will become the inputs for the second AND gate. Hence

$$X = (A.B).(C + D)$$

is the Boolean expression of the logic circuit.

EXERCISES

ESSAY TYPE QUESTIONS

1. State the various postulates of Boolean Algebra.
2. If A,B,C are any three Boolean Variables, prove from truth tables that the commutative and distributive laws hold good.
3. Prove the following theorem :

"The complement of the sum (OR) of two variables is equal to the product (AND) of the complement of the variables i.e., $\overline{X+Y}=\overline{X}.\overline{Y}$

4. State and prove De Morgan's Laws with the help of Truth Tables.

5. Explain the AND and OR operations along with their circuit diagrams.

SHORT ANSWER QUESTIONS

6. Prove the following using Boolean Algebra :

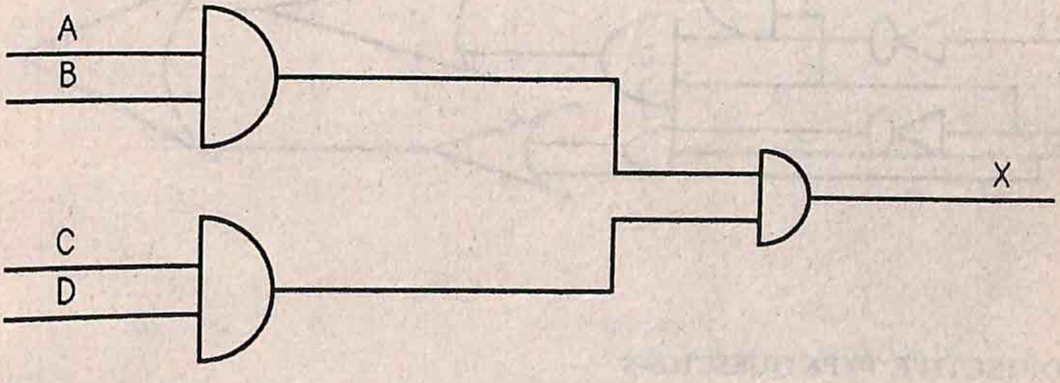
(i) $A + A = A$ (ii) $A.0 = 0$

(iii) $A.A = A$ (iv) $A + A.B = A$

(v) $\overline{\overline{A}} = A$ (vi) $A. (\overline{A} + B) = A.B$

7. Simplify the boolean expression $\overline{X}.Y + X.\overline{Y} + \overline{X}.\overline{Y}$ and draw a diagram showing the various gates.

8. Obtain the Boolean expression for the following logic circuit.



9. Prove the following :

(i) $X.Y + \overline{X}.Z + Y.Z = X.Y + \overline{X.Z}$

(ii) $X + X.Y = X$

10. Write a Boolean Expression that will have 1 as output when $X = 1$, $Y = 1$, $Z=0$ and 0 as the output in all other cases.

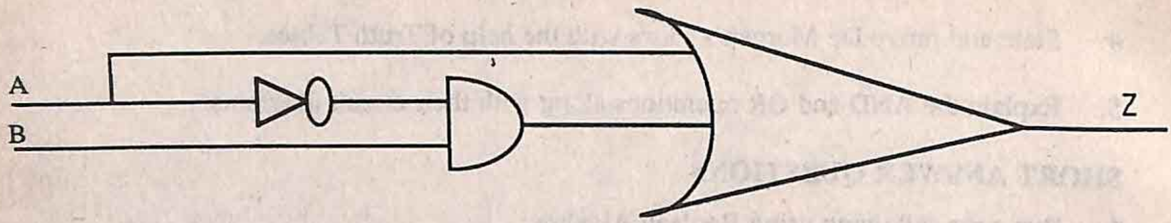
11. Define the following terms giving examples :

(i) Boolean Variables

(ii) Boolean Expression

(iii) Truth tables.

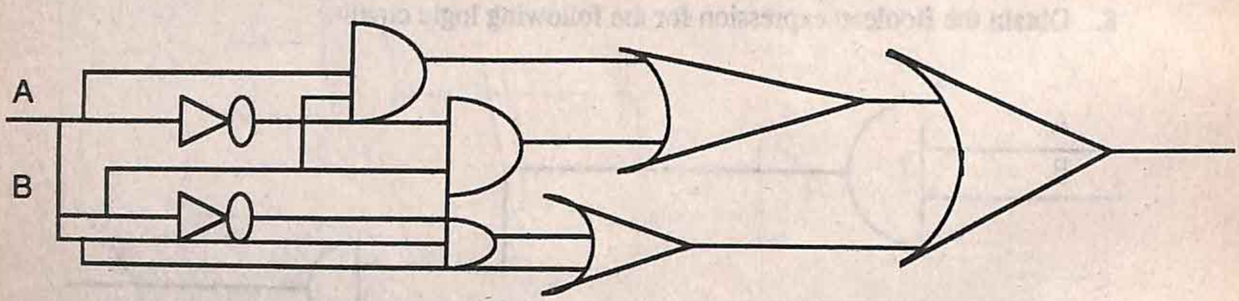
12. Complete the truth table for the following logic diagram :



13. Show how the following gates can be built using NAND gates.
    - (*i*) AND gate
    - (*ii*) OR gate

(Hint : A NAND gate is an Inverted AND gate *i.e.*, an AND gate followed by a NOT circuit)

14. Explain the output of the machine in terms of A and B. Show that this circuit is equivalent to a simple OR circuit.



OBJECTIVE TYPE QUESTIONS

15. Which of the following propositions are true if $X = 3$, $Y = 4$ and $Z = 5$
    - (*a*) $(X < Y)$ AND $(Y > Z)$
    - (*b*) $(X + Y) > (Y + Z)$
    - (*c*) NOT $(C > Z)$ OR $(Y \quad Z)$
    - (*d*) $(Z \quad X)$ AND (NOT $(Z < Y)$)

16. Draw truth tables for the following :
    - (*i*) $X.(Y + Z) + X.\overline{Y}$
    - (*ii*) $(X.(Y + \overline{Y}) + \underline{X}.(\overline{X}.\overline{Y}).\overline{Z}$
    - (*iii*) $A.(\overline{B} + C) + \overline{C}$
    - (*iv*) $W.X.Y.Z$
    - (*v*) $Y.Z + X.Z + X.Y + Y.Z.X$

17. Check the validity of the equations

$$(X+Y).(Y+Z).(Z+X)=(\bar{X}+\bar{Y}).(\bar{X}+\bar{Z}).(\bar{Y}+\bar{Z})$$

$$(X.Y)+(Y.Z)+(Z.X)=(\bar{X}+\bar{Y}).(\bar{Y}+\bar{Z}).(\bar{Z}+\bar{X})$$

18. Derive the boolean expresion from the following truth table and then simplify it.

(i)

A	B	C	D	Expression Value
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

(ii)

A	B	C	Function Value
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

19. Complete the following tables.

(i) $\overline{X+Y}=\overline{X}.\overline{Y}$

X	Y	X + Y	$\overline{X}$	$\overline{Y}$	$\overline{X+Y}$	$\overline{X}.\overline{Y}$
0	0	—	1	—	—	1
—	1	1	1	0	0	—
1	—	—	0	1	—	0
—	—	1	0	—	—	—

(ii) $\overline{(X.Y)}=\overline{X}+\overline{Y}$

X	Y	X.Y	$\overline{X}$	$\overline{Y}$	$\overline{X.Y}$	$\overline{X}+\overline{Y}$
0	—	0	—	1	1	—
0	1	—	—	0	—	1
—	0	—	0	—	1	—
1	—	—	0	0	—	0

(iii) $A.(\overline{B}+C)+\overline{C}$

A	B	C	$\overline{B}$	$\overline{B}+C$	$A.(\overline{B}+C)$	$\overline{C}$	$A.(\overline{B}+C)+\overline{C}$
0	0	0	—	—	0	1	—
0	—	1	1	1	—	—	0
0	1	—	—	0	0	1	—
0	—	1	0	—	—	0	0
1	—	—	1	1	—	1	1
1	0	1	1	—	—	0	—
1	1	0	0	0	0	1	1
1	1	—	—	1	1	0	—

9

INTRODUCTION TO BASIC

INTRODUCTION

BASIC (Beginner's All Purpose Symbolic Instruction Code) is the most popular and conversational programming language. It was initially developed at Dartmouth collage around mid-60's by J.G. Kemeny and Thomas Kurtz. Its own simplicity and increasing use of interactive computers soon made BASIC quite popular. Further, the language was found quite attractive for small, micro-processor-based computers.

Almost every manufacturer supplies BASIC language with his computer system. The BASIC language differs in minor respects from computer to computer. The disadvantage of this is that the programs written for one machine cannot be run on another without modification. At times, some facilities of BASIC may be absent. Fortunately, these differences are minor and can be easily learned. Whenever possible we ourselves will indicate points where differences may be present.

An interactive language like BASIC is usually interpreted by a computer and programs are not translated or compiled into machine language. The software which enables a computer hardware to understand BASIC is called BASIC interpreter.

In BASIC, the instructions are similar to algebraic formulae involving basic operations.

A programming language, like any other language, has an alphabet, a vocabulary and rules of grammar. The grammar rules, which are also called syntax rules, are used to construct commands or statements to be given to the computer. The programming languages are languages with very rigid rules of grammar. A seemingly trivial error is also not tolerated. The translator itself can point out errors in the commands, if any. Every command formulated by us will have a unique interpretation. Learning a programming language is learning the rules of syntax and interpretation (also called semantics) of commands.

The important features of BASIC can be summarized as below :

1. Simplified grammar.
2. Suitable for conversational programming.
3. Facility for manipulation of strings.
4. Dynamic program debugging.
5. Ability to carryout arithmetic operations on strings.
6. Suitable for both scientific and business applications.

A BASIC SYSTEM

One may access BASIC on a large time-sharing computer or on a small system. In both cases, our access is through a terminal consisting of a keyboard and a VDU screen. The terminal will be used for keying in programs, entering data and displaying results. It will be connected to the computer as a peripheral device. A large system will also have magnetic tapes, disk drives and printers–whose features are already discussed in earlier chapters.

The terminal will be the most important device for BASIC programming. The key board will have keys for digits, letters and various special symbols like +, —, /, etc. There will be a space bar at bottom which acts as a key for blank symbol. Some keys on the keyboard will have two symbols written on them. Depressing such keys will enter the lower symbol into the screen. To enter upper one, we have to press the SHIFT key, simultaneously with that key. Like typewriter, the key board also has TAB key for setting up markers and positioning cursor.

A keyboard will also have a few special purpose keys. The keys on your key board may be different from the ones given below. Find out which keys on your keyboard do similar functions.

RETURN key : is used to indicate the end of a line of program or data. The line typed by us will be taken in by the computer when this key is pressed.

→ Key : is used for moving the cursor to right.

← Key : is used for moving the cursor to left.

↑ Key : is used for moving the cursor up.

↓ Key : is used for moving the cursor down.

REPEAT key : Pressing this key simultaneously with another will repeatedly enter symbol of the later key.

ESCAPE key (ESC) : is used generally in editing a previously typed line and indicating end of editing.

CONTROL key (CTRL) : is generally used for controlling the ongoing work of the computer. It is usually followed by depressing a few more keys that indicate what we want the computer to do next for us.

There may be a few more keys on your terminal. Try to familiarize yourself with their functions by consulting system manuals :

MODES OF OPERATION AND INTRODUCTORY EXAMPLES

A BASIC system can operate in one of the following two modes.

(i) **The command or immediate execution mode:** In this mode it accepts one command at a time and immediately executes the keyed-in command. This mode is also called calculator mode. After executing the current command, the command is lost, and the computer will be ready for next command. This, we can notice with the appearance of READY prompt or > prompt. This mode is useful for simple calculations. Here we do not first have to develop programs and then execute them.

(ii) **The program mode:** Here, the BASIC system will accept program statements one after another, without executing them. A separate command is then required to cause execution of the entered statements. The statements will be remembered by the system even after their execution. A simple mechanism is used to distinguish between these two modes. In both the modes the commands or statements are entered as one line at the terminal. The only difference between a command and program mode is that the statements in the latter are preceeded by line numbers.

Let us consider some examples to illustrate these modes.

Try typing the following line.

```
PRINT “YOUR NAME PLEASE”
```

After depressing the RETURN key the computer will respond by printing on the terminal

```
YOUR NAME PLEASE
```

and will again be ready for next command, i.e., the PRINT command prints the characters between quotations marks as they appear. PRINT is a word from BASIC and the PRINT command execution consists of printing whatever was asked for in the command. If the command is :

```
PRINT 20 + 128
```

The response will be

```
148
```

You can see that there are no quotation marks here. If there are quotation marks, then the response will be

```
20 + 128
```

So whenever some calculations have to be done, we should not use quotation marks. In this mode we can use more than one command as below :

```
LET R = 13.5
```

The command will be executed, but there will be no response on the terminal since this command does not cause any printing. The command will be lost, but its effect of setting variable R to the value 13.5 will remain. Next,

```
PRINT R*2
```

will produce

```
27
```

from the system. One can easily see in this mode, the computer does behave like a calculator. In program mode, every line starts with a line number. The computer will keep accepting our statements. Suppose we type the following :

```
10 PRINT “YOUR NAME PLEASE”
```

Even after pressing the RETURN key, no character appears on the screen. This is because of the line number at the left. This line number will be remembered, but is not executed. We can execute this by giving the following command

```
RUN
```

Now type RUN and press RETURN key. The above statement will now be executed, unless we made some mistake in typing it (in which case, the computer will give error message) and the result of the print statement will appear :

```
YOUR NAME PLEASE
```

Now try the following program :

```
10    PRINT    “ ********”    (RETURN)
20    PRINT    “   ***   ”    (RETURN)
30    PRINT    “    *    ”    (RETURN)
RUN   (RETURN)
```

The following output will appear on the screen :

```
* * * * * * * *
     * * *
       *
```

A series of statements given in the program-mode makes up a program. We can run that program and store it on floppy using special commands provided in BASIC. Hence, we will be more interested in the program mode than in command mode.

ELEMENTS OF BASIC

In this section we introduce the preliminary rules of BASIC. You will find them quite simple to remember.

Character Set

The character set of BASIC includes alphabets, digits and special characters. The special characters include the following :-

=, :, (+ - * < > ↑ /) . $ % £ ' (blank)

The character set of any language defines the set of symbols to be used in constructing words, phrases and sentences of that language. In BASIC, programs are made up of statements and a statement of words and special symbols.

Constants

A constant is one that does not change its value during the execution of a program. There are two types of constants.

(1) Numeric constants
(2) String constants.

Numeric Constants

The numeric constants are numerical quantities. These may be integers or real numbers. The following are a few examples of integer constants.

120, –350, +963, 19785

The real constants can be written in two ways : a number with a decimal part and a fractional part, and with the exponent notation using E, that stands for 'multiplied by ten to the power'. The following are a few examples of real constants.

384.0 – 7.23, 6.12E + 5, 0.2345

The rules for writing BASIC constants are as follows :

- (*i*) a numeric constant can be preceded by + or — sign. In case no sign is used the constant is regarded as positive.
- (*ii*) E indicates exponential notation. Thus 3E + 5 means 3×10^5.
- (*iii*) The decimal point can occur only once inside the number.
- (*iv*) The exponent may also have a sign.
- (*v*) Most BASIC versions allow 8 or 9 significant figures in a constant.
- (*vi*) No special characters are allowed in a constant except decimal point.

String Constants

A sequence of characters is called a string constant. These include alphabets, numerals and special characters except quotation marks. A string constant must be enclosed in quotation marks but they do not form part of the string.

Strings are used to represent non-numeric information, such as names, addresses, days, etc. The following are a few examples of valid string constants :

" MOHAN DASS "
" PAY = "
" A 23578 "
" 245678 "

Note: 245678 is a numeric constant while "245678" is a string constant. No arithmetic operations can be performed on a string constant.

Variables

A quantity that can change its value during execution is called a variable. In a program we set different variable names for different items which we want to store separately. Variables are of two types :

1. Numeric Variables
2. String Variables

Numeric Variable

A numeric variable can be any single letter of the alphabets followed by one of the numerals from 0 to 9. Though it varies from computer to computer, the length of a variable name can be upto 16 characters depending on the system used. But again it has to start with an alphabet followed by a combination of alphabets and numerals. The following are a few examples of Numeric Variables :

A, B, C1, D5, A0, etc.

Note that, a numeric variable should always start with an alphabet, *i.e.*, A2 is a valid numeric variable while 2A is an invalid numeric variable. Numeric Variables are used only for assigning numeric constants. Special characters are not allowed in the variables, *e.g.*, S. I. is invalid and SI is a valid numeric variable.

Example : For calculating simple interest we use the formula

SI = P*R*T/100

Where SI, P, R and T are numeric variables and 100 is a numeric constant.

String Variables

An integer variable name that ends in the symbol $ represents a string variable. The following are a few examples of valid string variables :

A$, B1$, C$

The following are invalid statements :

A = "NAME"
A$ = NAME

Because "NAME" is a string constant and A is a numeric variable. String constants cannot be stored in numeric variable or vice versa. Mathematical calculations also cannot be performed using string variables.

Arithmetic Operators

BASIC provides the usual arithmetic operations on numeric quantities, with symbols and meaning as follows :-

+	Addition
—	subtraction or negation
*	multiplication
/	division
**	exponentiation

The operands on which these operators work may be real or integer type. When performing arithmetic operation, an erroneous condition called **over-flow** may arise if the result of operation is too large to be held in one location of computer's memory.

Expressions

An expression is composed of operators and operands. An operand may be a variable or a constant. An expression indicates one or more operations to be carried out in a specific order. When an expression contains many operators (as in A + B* C/D), the order in which they are evaluated are given by the following rules :

(*i*) First the exponentation operations are carried out.
(*ii*) Multiplication and division are performed next with same priority. They are evaluated left to right in the order of their occurrence.
(*iii*) Finally addition and subtraction operations are performed in the order of their occurrence from left to right with same priority.
(*iv*) The evaluation of expression is done from left to right.

These rules are followed strictly as long as there are no parentheses. If there are parentheses, then the expression within the parentheses is evaluated first following the above rules.

Now let us consider some examples :

A + B*D

Here, B*D is evaluated first and the addition is performed next.

A/B + C—D*E

First A/B and next D*E are performed. C is then added to the result of A/B. Finally the result of D*E performed earlier is subtracted from the result of Addition.

(A + B) *D

First A +B is done and the result is multiplied by D.

PRINT Statement

We saw some examples in the previous sections how a PRINT statement is used on computer. Not let us study PRINT statements in detail. The general format of the PRINT statement is <line no.> PRINT <Constant> or <Variable>. To separate more than one constant or variable a comma or semicolon can be used. The difference between the two is that in case of comma next item will be displayed from next zone. Where as semicolon starts displaying from the next position onwards. The screen of the VDU is generally divided into 4 zones. In case of 40 character screen, 1st zone starts from 1st column, 2nd from 11th, 3rd from 21st and 4th from 31st. Suppose we want to print a heading having various titles as ROLL NO, NAME and ADDRESS. Since we know that use of comma as separator will make it possible for us to print the various titles from different zones, the following PRINT statement will print the heading as desired.

```
10 PRINT “ROLL NO”, “NAME”, “ADDRESS”
RUN
```

The above statement will display ROLL NO starting from 1st column, NAME from 11th column and ADDRESS from 21st column. The output will look like :—

```
Column
nos. 1           11        21
     ROLL NO     NAME      ADDRESS
```

Another way to print the above heading is by using semi-colon as separator. Now let us try to print the same heading using the following statement :—

```
10 PRINT “ROLL NO”; “NAME”; “ADDRESS”
RUN
```

In the output no blank is left between ROLL NO, NAME and ADDRESS.

```
Column 1
     ROLLNONAMEADDRESS
```

The output corresponding to each PRINT statement begins on a new line. For example the following program will print the result in two different lines:

```
10 PRINT “INDIA”
20 PRINT “ IS A BIG COUNTRY”
RUN
```

You will find the display as

```
INDIA
IS A BIG COUNTRY
```

Now let us modify line number 10 and 20 as below and add a new line.

```
10 PRINT “INDIA”;
20 PRINT “ IS A BIG COUNTRY”;
30 PRINT “I LOVE IT”
```

There are three different PRINT statements. As per our previous experience the output should be in three different lines. But when you RUN this program you will find that the result will be in a single line as below :

```
INDIA IS A BIG COUNTRY. I LOVE IT.
```

This is because of the semicolon appearing at the end of the PRINT statement in lines numbered 10 and 20. The number of blank spaces between the results of different PRINT statements will be same as discussed above, that is if there is a comma then the result of the next PRINT statement will be printed in the next zone and if there is a semicolon at the end of the present PRINT statement then there will be no blank space between the result of the current PRINT statement and the next PRINT statement.

A Few Rules:

1. The output corresponding to each PRINT statement begins on a new line and single PRINT statement may have more than one output line.
2. Strings in the list of PRINT statement

must be enclosed in quotation marks and they will appear on the output line as it is.

3. If the list in the PRINT statement contains no items, then a line will be skipped, i.e., a blank line will appear.
4. If a comma or semicolon appears after the last item of a PRINT statement then the output corresponding to the next PRINT statement will appear on the same line.

TAB Function:

The TAB function is used in the PRINT statement to indicate printing from a particular column for example,

```
10 PRINT TAB (10); "INDIA";
TAB (20); "PAKISTAN"
RUN
```

Will output the result as follows :

```
Column no. 1 :      11         21
                    INDIA      PAKISTAN
```

This function is useful to format the output in the desired form.

ASSIGNMENT: The LET Statement

The result of computations in an expression can be stored as a value of some variable by using the assignment statement. The LET statement allows us to assign a value to the variable. The general format of LET statement is as follows:

<LINE NUMBER> LET <Variable> = <Constant> or <Variable> or <expression>

Suppose you are given a problem to solve, in which you have to multiply two numbers and find the result. When you do this after calculating the result you have to store it in memory using some variable name to retrieve the answer later. For example, you are asked to multiply 5 by 7 and add the result to 10 and store it in some variable X. This can be written as :

```
10 LET X = 5*7 + 10
```

Similarly, if you want to assign a value 10 to the variable A, you have to write this as

```
20 LET A = 10
```

Similarly a string value can be assigned to a string variable, say, A$ is the variable and we want to store INDIA in this variable. We will write this as

```
20 LET A$ = "INDIA"
```

Note : It is not possible to assign a value to a constant.

For example 10 Let 2 = 50 is an invalid statement.

The following are some valid LET statements used to assign numeric values to variables:

```
10 LET B = 5
15 LET SUM = 10
11 LET P1 = (A + B + C)/3
```

Here 5,10 and (A + B + C)/3 are assigned to the variables B, SUM and P1 respectively. In line number 11, the calculations are done using the values of variables A, B and C and the result calculated is stored in the variable P1.

The following are a few example of assigning string values using LET statement.

```
10 LET N$ = "MY NAME IS
              SHAHIDA"
20 LET B$ = "I LOVE INDIA"
30 LET C$ = "SINGUPALEM"
40 LET D$ = "HELLO SIR"
```

In the above example MY NAME IS SHAHIDA, I LOVE INDIA, SINGUPALEM, HELLO SIR are string constants enclosed in quotation marks that are stored in computer memory with addresses N$, B$, C$ and D$.

Suppose we have a small program :

```
10 PRINT "NAME", "PERCENTAGE"
20 LET A = 10
30 LET B = 20
40 LET C = 30
50 LET N$ = "SHAHIDA"
60 LET D = (A + B + C)/150 *100
70 PRINT N$, D
80 END
RUN
```

In this program A, B and C are the marks in 3 subjects, each out 50. N$ is a string variable having the value SHAHIDA. The percentage is calculated and stored in D. The output of the above program will be

```
NAME              PERCENTAGE
SHAHIDA                40
```

Let us see what will happen in the following assignment statement :

```
10 LET A = A + 1
```

The values will be transferred from right-hand-side to left-hand-side, that is first right-hand-side is calculated and the result will be assigned to left-hand-side. So first A + 1 will be calculated, that is computer will take the contents of memory location having symbolic name A and add 1 to that and this new value will be stored in variable A itself. Now the previous value of A is lost. This particular statement is very helpful in counting.

LINE NUMBERING

Every line of a BASIC program starts with a number. This can be any number starting with 1 and ending with 99999. The upper limit of the line number varies from computer to computer. A line number is separated from the rest of the statement by one or more spaces. The line numbers play vital role in BASIC. We have already seen one of them, that is, it distinguishes between immediate and program mode. The other important benefits rules of line numbering are as follows :

(i) It is equivalent to numbering of steps in the description of an algorithm. As it uniquely identifies the position of a statement within a program, a line number is used to divert the flow of execution to any statement of the program.

(ii) The line numbers are used in BASIC to arrange the statements in increasing order of their line numbers. The sequential flow of execution proceeds in the increasing order of line numbers. Since BASIC automatically orders the statements, we can enter our statements in any arbitrary order. To illustrate, this, let us consider a program consisting of the following statements :

```
10 LET A = 12.5
20 LET B$ = "PAYROLL"
30 LET C = A*2/10.5
```

These could have been entered in the following sequence without affecting the results.

```
20 LET B$ = "PAYROLL"
10 LET A = 12.5
30 LET C = A*s/10.5
```

An advantage of this facility is that even if a statement is omitted by oversight it can be typed at a later stage by giving it a suitable line number that will put it into the desired position. That is one important reason why we use line numbers with a difference of 5 or 10 instead of using consecutive line numbers such as 1, 2, 3, etc.

(iii) Any statement, irrespective of its position in the program, can be easily modified afterwords just by using its line number. Suppose the value of A in the above program is 15 instead of 12.5. Without retyping any other statement, we can change this statement by retyping

```
10 LET A = 15
```

If we want to delete a particular line we have to type simply that line number and press return. Suppose we want to delete the line 20 LET B$ = "PAYROLL". Then we have to type simply 20 and press return. After this the entire line will be deleted from the computer's memory.

EXERCISES

ESSAY TYPE QUESTIONS

1. What is the full form of BASIC ? Discuss the important features of this language.
2. What are the two modes in which BASIC system can operate ? Explain them.
3. What are constants and variables ? Explain their types along with some examples.
4. What is an arithmetic expression ? What are the rules for forming an arithmetic expression ?

SHORT ANSWER QUESTIONS

5. Write down some important rules for writing a PRINT statement.
6. What is a TAB function ? Explain in brief giving examples.
7. What is a LET statement ? Give its general format.

OBJECTIVE TYPE QUESTIONS

8. Fill in the blanks :
   (1) The quantities that do not change while executing the programs are called ....
   (2) The constants are of .... types : .... constants and .... constants.
   (3) A quantity that may change its value during the execution of a program is called a ....
   (4) .... statement is used for displaying results on the screen.
   (5) Full form of BASIC is ....
   (6) .... statement is called the Assignment statement.
   (7) Numeric constants can be stored in .... variables only.
   (8) In PRINT statement .... used as separator causes the value to be printed in next zone.
   (9) .... function is used to control spacing in the output.
   (10) "3428" is a .... Constant.

9. State whether following statements are 'True' or 'False'.
   (1) Commas are allowed in a numeric constant.______________
   (2) Quotation marks do not form a part of the string constant. ____________
   (3) A variable represents memory location in computer's memory.____________
   (4) Semicolon causes the value to be printed in the next zone. ______________
   (5) LET 2 = X is a valid statement. _________________

10. State whether the following string constants are valid or invalid. If invalid write it in valid form :

	STRING CONSTANT	VALID/INVALID	CORRECT FORM
(i)	VIVEK	....	....
(ii)	"23 - C New Delhi"	....	....
(iii)	2345	....	....
(iv)	7 MARCH	....	....
(v)	"5 Kg"	....	....
(vi)	"10 + 20"	....	....

11. Assign the following values to appropriate variables given next to them :

(i) INDIA IS MY COUNTRY, A$ ....

(ii) A - 132, B$ ....

(iii) 246, N ....

(iv) COMPUTER SCIENCE, C$ ....

(v) 14th NOV, 88, D$ ....

12. State whether following numeric constants are valid or not. If invalid write it in correct form :

	CONSTANT	VALID/INVALID	CORRECT FORM
(i)	+ 946	....	....
(ii)	— + 46.4	....	....
(iii)	20,000	....	....
(iv)	1.32—E5	....	....
(v)	—5E + 2	....	....

(vi)	—460	....	....
(vii)	92—	....	....
(viii)	8.69E—5	....	....
(ix)	1.24	....	....

13. State whether the following variables are valid or not. If invalid write it in correct form :

	VARIABLES	VALID/INVALID	CORRECT FORM
(i)	A1	....	....
(ii)	NAME$	....	....
(iii)	S.I	....	....
(iv)	AREA	....	....
(v)	NET-PAY	....	....
(vi)	PRINT	....	....
(vii)	2B	....	....
(viii)	N1$	....	....
(ix)	$ N	....	....

14. State whether the following statements are correct or not.

		CORRECT/ INCORRECT
(i)	PRINT 230—AC	....
(ii)	PRINT 2469; "AJAY"	....
(iii)	PRINT, "458"	....
(iv)	PRINT A, B	....

(v) PRINT TAB(20) "2-A DELHI",TAB(5); X ....

(vi) PRINT "N $", A ....

(vii) PRINT "AREA=A"(Where A has value 6) ....

(viii) PRINT 32,000 ....

(ix) "PRINT" N$, Y$; X ....

15. Write the output of following statements :

(i) 10 PRINT "INDIA IS MY COUNTRY"

....

(ii) 20 PRINT TAB (3); "COMPUTER"; TAB (15); "SCIENCE"

....

(iii) 35 PRINT TAB (10); "AMOUNT =";
40 PRINT TAB (25); X; "RS" (Where X = 20)

....

....

(iv) 10 PRINT "S. NO.", "NAME", "AGE"

....

(v) 25 PRINT C; D; E; F (Where C = 2, D = 1, E = 4, F = 8)

....

16. Write a BASIC expression that corresponds to each of the following algebraic formulae:

	Algebraic formula	**BASIC Expression**
(a)	$3X + 5$	....
(b)	$i + j - 2$	....
(c)	$X^2 + Y^2$	....

(d) $a\,(b + c) - 1\,m$ ....

(e) $\frac{a}{b}+\frac{c}{d}$ ....

(f) $\frac{u+v}{k-1}$ ....

(g) $4\,t + \pi\, r^2$ ....

17. Write LET statement for each of the following situations :

(a) Assign a value of 2.54 to the variable C.

....

(b) Assign the value represented by N to the variable N1.

....

(c) Assign the string JAN 31 to the variable S$.

....

(d) Increase the value assigned to the variable C7 by 0.01.

....

(e) Assign the value represented by the formula (A2+B2 + C) to the variable F.

....

(f) Decrease the value assigned to variable K by 2.

....

18. Consider the statement :
25 LET P = Q. 4
if Q = 2, what value will be assigned to P?

....

....

19. Show how the output will appear in each of the following situations:

Where A = 0. 0000062 P1 = 743.08
B = 27.5*10 P2 = 0.00987
C = —1000 Q = 48.8*10
P = 2,770,543

Hint : Do these problems on the computer and check your answers.

(a) 100 PRINT A; B; C; P; P1; P2; Q

(b) 110 PRINT A, B, C, P1, P2, Q

(c) 120 PRINT A + B + C, P/Q, P1/P2

(d) 130 PRINT N$, P$, Q$

Where N$ = "RAM"
P$ = "DEC 29, 1963"
Q$ = "ELEVEN O'CLOCK"

□ □

10

SYSTEM SOFTWARE

INTRODUCTION

So far, we have learned elementary rules of BASIC and we know how BASIC can be made to perform computations using its LET statement. We must study a few more statements before we can write complete and more meaningful programs. In this chapter, we consider decision-making statements, the control structuring statements and the statements for input and output. We will indicate the rules for constructing these statements, i.e., rules of syntax, meaning or semantics of these statements and the actions performed by BASIC.

REM Statement

The word REM stands for REMarks. A REM statement is used to include explanatory comments in the program, which are very useful to explain what a group of statements achieve and what a variable stands for. The general format of REM statement is

<Line number> REM <any text>

It is a non executable statement and it doesn't affect the program results in any way. The REM statements help the programmer and others in reading and understanding programs and this statement can be used anywhere and any number of times in a program. The following examples will explain the use of REM statements :—

```
10 REM     Program for finding sum of two
           numbers
20 REM     This program was written by
           Naveen on 10-2-1987
30 LET     A = 35
40 LET     B = 45
50 LET     S = A + B
60 PRINT   S, A, B
70 END
```

In the above program REM statements provides information regarding what is being done in the program, programmer's name and the date on which it was written.

```
10 LET     X = 25
20 LET     Y = 35
30 REM     SUM OF TWO NUMBERS
40 LET     S = X + Y
50 PRINT   S
60 REM     DIFFERENCE  OF  TWO
           NUMBERS
70 LET     D = X — Y
80 PRINT   D
90 END
```

As shown in the above example REM statement can be used anywhere in the program and any number of times. In this example REM statement is used at two different places to define two different processes being done within the program.

STOP Statement

The STOP statement is used to stop the execution of the program temporarily. If the program execution is stopped by the STOP statement, we can still continue the execution from that statement onwards. The general format of the STOP statement is

<Line Number> STOP

There may be more than one STOP statement in a program and it can be placed anywhere in the program.

In the following example :

```
10 LET     X = 5
20 LET     Y = 15
30 REM     SUM OF TWO NUMBERS
40 LET     S = X + Y
50 PRINT   S
60 STOP
70 REM     DIFFERENCE  OF  TWO
           NUMBERS
80 LET     D = Y — X
90 PRINT   D
100 END
```

The execution will stop after printing the sum. If we want to get the difference also, then by using a command like CONTINUE (which will differ from computer to computer) we can initiate the execution again. Then the computer starts execution from line number 70 onwards.

END Statement

The end of a program is always indicated by the END statement. When this statement is executed the execution of program will stop permanently. The format of the END statement is

<Line number> END

The END statement should always be the last statement of the program. Let us see what will happen if there is an END statement in the middle of a program.

```
10 LET     X = 5
20 LET     Y = 15
30 REM     SUM OF TWO NUMBERS
40 LET     S = X + Y
50 PRINT   S
60 END
70 REM     DIFFERENCE  OF  TWO
           NUMBERS
80 LET     D = X — Y
90 PRINT   D
100 END
```

After printing the sum the execution will stop permanently because of the END at line number 60 and it is not possible to continue the execution of the program further as in STOP statement. So, the program will never print the difference.

Hence there should be only one END statement in the program and it should be the last statement of the program.

INPUT Statement

Previously we used LET statement to assign a value to some variable. In this case we can assign a value to only one variable at a time. So we must have as many lines as the number of variables we want to define. But this is not an efficient way of assigning values to variables. Another disadvantage is that we have to change our program every time we want to change the data. To solve these problems, INPUT statement is used to assign values to variables. INPUT statement takes the values from the keyboard when the program is executed. The advantage of INPUT statement over LET statement is that we can assign values to more than one variable in one statement and we can change these values for every run. Hence, we can give various sets of data to obtain different results in different runs.

The general format of the INPUT statement is :

<Line number > INPUT <List of

variables>

If there are more than one variable in the list, then they should be separated by commas. During the execution of program, when the control comes to INPUT statement, a question mark appears on the screen and until and unless we give the value here and press RETURN, control does not move to next statement. The value given here should match with the type of variable, i.e., if it is string variable, the value entered should be string constant and if numeric then value has to be numeric constant. If we are using more than one variable then these values have to be given in the same order of variables and should be separated by commas. If we provide more values than the number of variables then the extra values will be ignored.

Now let us take some examples. Suppose we want to add two numbers. Using LET statement the program can be written as :—

```
10 LET    A = 15
20 LET    B = 25
30 LET    C = A + B
40 PRINT  C
50 END
```

If we run this program ten times, it will print the same value, i.e., 40, everytime.

Now, if we want to add 10 sets of data, then we have to change the first two lines every time. If we use INPUT statement to assign values to A and B, then the program will look like :

```
10 INPUT  A, B
20 LET    C = A + B
30 PRINT  C
40 END
```

When we execute this program, a question mark will appear on the screen. Here we have to provide the values of A and B separated by comma and then press RETURN. These values will be stored in memory in the cells named A and B. After adding these values the result will be stored in C and then it will be displayed on the VDU. This program can be executed again and again, and we can give different values of A and B each time.

```
RUN       <RETURN>
? 12,5    <RETURN>
17
RUN       <RETURN>
? 25,250  <RETURN>
275
```

In INPUT statement we can also give the messages with the variables to identify them. In the above case INPUT A,B will simply display the prompt ? on the screen. In this case one might not know what values are to be given and whether they are string or numeric variables. And also, we do not know how many values have to be given. We can modify the above program, with the help of messages in INPUT statement to overcome the above difficulties as below.

```
10 PRINT "TO ADD TWO NUMBERS"
20 INPUT "FIRST NUMBER"; A
30 INPUT "SECOND NUMBER"; B
40 LET   C = A + B
50 PRINT "SUM OF GIVEN NUMBERS
         IS"; C
60 END
```

The message in the INPUT statement should be enclosed in quotation marks and the message and variable name should be separated with a semicolon. After executing the above program the following display will come on the screen :

```
TO ADD TWO NUMBERS
FIRST NUMBER ? 10 (The number is to be
given by user) <RETURN>
SECOND NUMBER ? 15 <RETURN>
SUM OF GIVEN NUMBERS IS 25
```

GOTO Statement

The GOTO statement is the simplest of all control statements and is used for unconditional transfer of execution from one part of the program to the other. Thus, it breaks the sequential flow of execution and diverts the control to some other place in the program. The general format of GOTO statement is :

```
<line number>        GOTO  n
```

Where *n* is the line number of a statement in the program. For example, the following GOTO statement will cause the statement

```
150 GOTO 320
```

number 320 to be executed after statement number 150 leaving all the statements between 150 and 320 untouched. The statement numbered n may be below or above the GOTO statement. If the statement numbered n is below, then it is called a **forward jump**. It is described in the following example :

```
10 INPUT     A,B
20 INPUT     X,Y
30 GOTO      60
40 LET       X = X*Y
50 LET       Y = Y*Y
55 PRINT     X,Y
60 LET       X = A*Y
70 LET       Y = B*Y
75 PRINT     X,Y
80 END
```

In the above example, statements numbered 40, 50 and 55 will be skipped and control of execution will come to line numbered 60 after executing line numbered 30.

If the statement numbered n is above GOTO statement, then some statements will be repeated and the jump is known as **backward jump.** It is explained in the following example :

```
10 PRINT     "TO ADD TWO NUM-
              BERS"
20 INPUT     "FIRST NUMBER"; A
30 INPUT     "SECOND NUMBER" ; B
40 LET        C = A + B
50 PRINT      "SUM OF GIVEN NUM-
              BERS IS"; C
60 GOTO       20
70 END
```

In the above program when the control of execution comes to line numbered 60 it will be again transferred to line numbered 20. So the computer will go on asking for more values of A, B because the control will never come to END statement given in line 70. So it becomes an infinite loop. To stop the execution of this program one has to press RUN/STOP or ESCAPE or some other key on the keyboard depending on the computer.

Now let us write some basic programs using the commands we have learnt so far.

Problem 1

Calculate the wages of one employee assuming that the wage rate per hour and number of hours worked are known.

```
10 REM        CALCULATING WAGE
              OF ONE EMPLOYEE
20 REM        R IS THE RATE IN RS
              PER HOUR
30 REM        N IS THE NUMBER OF
              HOURS WORKED
40 REM        W IS THE TOTAL WAGE
50 LET        R = 15
60 LET        N = 25
70 LET        W = R*N
80 PRINT      "RATE = RS." ; R; "PER
              HOUR"
90 PRINT      "TOTAL WAGE"; W;
              "RS."
100 END
```

In the above program lines numbered 10 to 40 are REMark statements to give comments on the program. In lines numbered 50 and 60, values are assigned to variables R and N respectively.

Line numbered 70 calculates the total wage and the result is assigned to W. Lines numbered 70 to 90 display the results in the following format :

```
RATE = RS. 15 PER HOUR
HOURS WORKED = 25
TOTAL WAGE = 375 RS.
```

We can generalize the above program using INPUT statement to calculate the wages of more than one employee.

```
10 REM      CALCULATING WAGES
20 REM      R IS RATE PER HOUR
30 REM      N IS THE NUMBER OF
            HOURS WORKED
40 REM      W IS THE TOTAL WAGE
            OF ONE EMPLOYEE
50 INPUT    "RATE", R
60 INPUT    "NO.  OF  HOURS
            WORKED"; N
70 LET      W = R*N
80 PRINT    "RATE = RS"; R "PER
            HOUR"
90 PRINT    "NO.  OF  HOURS
            WORKED ="; N
100 PRINT   "TOTAL WAGE ="; W;
            "RS."
105 GOTO    50
110 END
```

PROBLEM 2

To print a given string i.e.,"IS AN INTELLIGENT PERSON" with different names.

```
10 LET      S$ =  "IS AN INTELLI-
            GENT PERSON"
20 INPUT    "NAME"; N$
30 PRINT    N$ ; S$
40 GOTO     20
50 END
```

PROBLEM 3

To calculate the area of circles with different radii.

```
10 INPUT    "RADIUS"; R
20 LET      AR = (22/7)*R*R
30 PRINT    "AREA OF CIRCLE =";
            AR
40 GOTO     10
50 END
```

EXCHANGING VALUE OF TWO VARIABLES

Suppose there are two variables A and B with the values 4 and 6 respectively, i.e.,A=4 and B = 6 and we want to exchange the values of these variables so that A = 6 and B = 4. To do this we follow the following steps :

1. Add A and B and store the value in A.
2. Subtract B from A and store the result in B.
3. Subtract the new value of B from the new value of A and store the result in A.

Now you will find that the values of A and B are exchanged.

PROGRAM	Step-by-Step solution
10 PRINT A,B	1st — A = 4, B = 6
20 LET A = A + B	2nd—A=4+6=10
30 LET B = A—B	3rd—B=10—6=4
40 LET A = A—B	4th—A=10—4=6
50 PRINT A, B	5th — ANS — 6,4
60 END	

DECISION MAKING : IF THEN Statement

In the IF-THEN statement, we evaluate a condition, and if it is found to be true, we can direct execution to some other part of the program. If the condition is found to be false, then the execution proceeds sequentially to next statement. Thus we have a conditional transfer of control. The general format of the IF-THEN statement is

```
<line no>  IF  <test condition>
THEN n
```

Let us consider an example

```
10   IF        A>B    THEN 100
20
.
.
.
100
.
.
.
```

If A is greater than B then control will go to line numbered 100 and the execution will continue sequentially from 100 unless it is disturbed. If A is not greater than B then control will move to the next line, i.e.,to line numbered 20.

In the test condition one of the following comparison or relational operators are used:

=	Check for equality
<	Check for less than
>	Check for greater than
<=	Check for less than or equal
>=	Check for greater than or equal
<>	Check for not equal to

The method of comparing two strings consists of comparing them character by character from left to right.

```
10      LET    A$ = “ABC”
20      IF     A$ = “ABCD” THEN 50
.
.
.
50
```

In the above example A$ will be compared with the string ABCD. First, the first character i.e.,A of the both will be compared. As these two are equal then the second character, i.e.,B of the string are compared. After comparing C, there is no character in A$ to compare with D of the other string. So condition yields a false result and the control moves to the next line.

Now let us see where this condition is useful in programming. Let us take examples of calculating area of circles of different radii and the problem of calculating wages for more than one employee.

The program for calculating area of circles, using INPUT statement is

```
10 INPUT  “RADIUS”; R
20 LET     AR = (22/7) *R*R
30 PRINT  “AREA OF CIRCLE = ”; AR
40 GOTO10
50 END
```

Because of the GOTO statement in line number 40, the control will never come to the last statement i.e.,END statement and it becomes an infinite loop. So the program will never terminate. To overcome this, we modify the above program using IF-THEN statement as below :

```
10  INPUT   “RADIUS”; R
20  LET      AR = (22/7)*R*R
30  PRINT   “AREA OF CIRCLE =”;AR
40  PRINT   “DO YOU WANT MORE
             PROBLEMS”
50  INPUT   Y$
60  IF       Y$ = “YES” THEN 10
70  END
```

In this program, after printing the area AR, the computer will display “Do you want more problems” with a question mark which is because of the INPUT statement at line number 50. Now you have to respond to the question mark. Your input value will be compared with YES in the IF-THEN statement at line number 60. If the result is True then only the control will move back to line number 10. Otherwise the control will move to the next statement which is END and the execution will be terminated.

Similarly in the example of wages also we shall be able to terminate the program if we use IF-THEN statement.

```
 10  REM       CALCULATING WAGES
 20  REM       R IS THE RATE IN RS.
               PER HOUR
 30  REM       N IS THE NUMBER OF
               HOURS WORKED
 40  REM       W  IS  THE  TOTAL
               WAGES
 50  INPUT     "RATE" ; R
 60  INPUT     "NUMBER  OF  HOURS
               WORKED"; N
 70  LET       W = R*N
 80  PRINT     "RATE = Rs.;  "R"; "PER
               DAY"
 90  PRINT     "DAYS WORKED"; N
100  PRINT     "TOTAL WAGES ="; W;
               "RS."
110  PRINT     "DO  YOU  WANT  TO
               CONTINUE"
120  INPUT     Y$
130  IF        Y$ = "YES" THEN 50
140  END
```

READ-DATA Statement

Many programs require a large number of data items to be entered into the computer. Though this can be done with the help of LET and INPUT statements, it becomes very cumbersome if something has to be done a number of times. This can be made more conveniently by the use of READ-DATA statement when we donot want to change the data again and again.

In READ statement, the variables are defined just like in INPUT statement, i.e.,if there are more than one variable then they are separated by commas. The general format of this statement is:

<line number> READ

<list of variables>

The list of variables can be Numeric or String or both. Suppose we want to assign the values to A$, B and C. Then the READ statement will be

```
15 READ  A$, B, C
```

Upon execution of this statement, the computer will obtain values for these variables A$, B and C, but in a different way than INPUT statement. Here we specify the values as part of program itself through DATA statement. A DATA statement contains one or more values separated by commas. There can be more than one DATA statement and the DATA statement can be anywhere in the program. Generally the DATA statements are placed at the end of the program for convenience. One can place DATA statement in the middle or beginning of the programs as well. The string values in DATA statement corresponding to the string variables in READ statement should be in quotes. Data statement is also a **non-executable statement.** Let us consider the example of READ statement given above.

```
 15    READ A$, B, C
        :
100    DATA "DELHI", 10,15
```

Now the string value DELHI will be assigned to A$ and the numeric values 10 and 15 will be assigned to B and C respectively. If we write the above DATA statement as :

```
100 DATA 10,15,  "DELHI"
```

then a type mismatch error will occur during execution because the first variable in READ statement is a string variable and the first value in the DATA statement is a numeric value.

The following example will describe the use of multiple READ-DATA statements.

```
 40  READ A,B,C
 50  READ P$, Q$
 60  READ F1, F2, F3
      :
      :
      :
210  DATA 5,— 2,11, "VBA"
220  DATA "RAO", —8, 0,10
```

These statements cause the numbers 5, —2 and 11 to be stored in A, B and C respectively and the strings VBA and RAO to be stored in P$, Q$ and the numbers —8, 0 and 10 to be assigned to variables F1, F2, and F3.

Let us take another example where READ-DATA statement is used.

To find the Average marks in five subjects.

```
10 REM      PROGRAM FOR FIND-
            ING AVERAGE MARKS
20 READ     A, B, C, D, E
30 LET      X = (A + B + C + D + E)/5
40 PRINT    X
50 DATA     10, 15, — 10,6,100
60 END
```

ARRAY PROCESSING IN BASIC

Upto now we have been using the same variable name to assign many sets of values to the variables. For example, in the previous sections when we calculated wages for different employees, the variables were taken as R, N and W. But if there are more than one employee, we have to assign values of the next employee to the same variables. Due to this the data for the first employee will be erased from the memory and if we want those values again during subsequent execution of the program we have to assign these values again to the variables. In turn the present value of the variables will be erased again. Suppose we want values of both the employees. Then we have to assign values to six different variables: three each for one employee. Suppose there are 1000 employees, then we require 3000 variables and it also becomes difficult to remember variables corresponding to different data values. A solution to the above problem is to define a single variable name for a particular type of data of all employees with suffixes in the brackets as below.

For the wages problem mentioned above we can define Rate per hour as R(I) for all the employees. As a result R (1) stands for rate for the first employee, R (2) stands for rate for the second employee and so on. Similarly we can define Number of hours worked as N (I) for all the employees. Here N (I) will stand for number of hours worked by the first employee, N (2) for number of hours worked by the second employee and so on. Similarly we can define wages for different employees as W (1), W (2) and so on.

As a result :

R (1), R (2), ..., R (1000)
N (1), N (2), ..., N (1000)
W (1), W (2), ..., W (1000)

will define the variables for all the 1000 employees, and we can retrieve any value, for example rate for a particular employee, say 632, by specifying it as R (632).

The names which refer to collection of values are called **array names** and the quantity written within the parentheses to select a particular value is called a **subscript**. In the above example R, N and W are the array names and the numbers from 1 to 1000 are called subscripts.

Each value in an array is called its **element,** that is if R(100) = 20 then 20 is called an element of the array. The total number of elements an array can have is called its **size.** In the above example, the size of the each of the arrays R, N and W is 1000. When we use arrays, it is necessary to tell the computer that we are using an array if the size of the array is greater than 10. This will be done using the following statement.

DIM Statement

DIM stands for DIMension. The general format of DIM statement is :

<line no> DIM <variable name with size of the array in parentheses>

The DIM statement should always be the first statement in the program. When the program is executed if there is any DIM statement in it, first the computer will reserve the required space in memory for the variables defined as array. We can give more than one array in the DIM statement, in which case different arrays will be separated by commas. For example, the DIM statement for the example of calculating wages will be :

```
10 DIM R (1000), N (1000), W (1000)
```

At present, using the concept of arrays looks very laborious. How this concept is simpler than the ones discussed so far will be clear from the following discussion.

FOR - NEXT Statement

Unlike human beings, computers never get bored doing the same thing over and over again. If some body wants to write his/her name 100 times, it will be a boring job to write PRINT statement 100 times. This can be done by using a GOTO statement instead of writing a PRINT statement again and again. Then also each time after printing the name the program should check whether the printing has been done the required number of times or not. For this we have to write a number of statements and the process becomes very slow. To overcome this, BASIC language provides a statement called FOR-NEXT. Whenever, some steps have to be repeated a number of times, we can put that module in FOR-NEXT loop.

The general format of FOR-NEXT loop is :

```
<Line no.>    FOR I = M TO N STEP K
.
.
.
<line no.>    NEXT I
```

where M is the initial value and N is the final value of the variable I. In the beginning the initial value M will be assigned to the variable I. Then it will execute all the lines until NEXT I, at which the value of I will be incremented by K. Then the new value of I will be compared with N. If it is less than or equal to N then the execution will be repeated starting from the next line of FOR statement and will contine upto next I statement. The process will be repeated until the value of I exceeds the value N in which case the control will move to the next line after NEXT statement. If STEP K is not mentioned, then the value of I will incremented by 1 every time.

Consider the following program

```
10  FOR I = 1 to 100
20  PRINT "RAJESH"
30  NEXT  I
```

This program will print the name **RAJESH** 100 times.

Instead of printing same name 100 times, suppose we want to print 100 different names. For this we will make use of the concept of arrays. As the size of the array is more than 10, we have to tell the computer to reserve memory space for 100 names through DIM statement. The program will be as below :

```
10  DIM      N$ (100)
20  REM      N$ REPRESENTS
             ARRAY FOR 100
              NAMES
30  FOR      I = 1 TO 100
40  READ     N$ (I)
50  PRINT    N$ (I)
60  NEXT     I
70  DATA     ........., ............,      ............
80  END
```

Here at line number 70 in the DATA statement we have to provide 100 names.

FURTHER EXAMPLES

Example 1.

To calculate the average marks of 50 students.

```
10  DIM     M (50)
20  LET     TOTAL = 0
30  FOR     I = 1 TO 50
40  READ    M
50  LET     TOTAL = TOTAL + M
60  NEXT    I
70  LET     AV = TOTAL/50
80  PRINT   “AVERAGE =”; AV
90  DATA    12, 15, 17, 30, 25........
100 END
```

Example 2.

Calculate simple interest for 100 individuals and print the name of the person along with interest.

```
10  FOR     X = 1 TO 100
20  INPUT   “NAME”; N$
30  INPUT   “PRINCIPAL”; R
40  INPUT   “RATE OF INTEREST”; R
50  INPUT   “TIME”; T
60  LET     I = P*R*T/100
70  PRINT   N$
80  PRINT   “SIMPLE INTEREST =”; I
90  NEXT    I
100 END
```

Example 3.

Print the multiplicationTable of 5. In this program the output should be like

```
5 × 1 = 5
5 × 2 = 10
5 × 3 = 15
5 × 4 = 20
```

and so on

```
10    FOR  I  = 1 TO 10
20    LET  M = 5*I
30    PRINT “5  * ” ;I;“=”;M
40    NEXT  I
50    END
```

If we want to generalize the above program, i.e., to print the multiplication table of any number, then the modifications will be as follows :

```
10 INPUT   “THE NUMBER FOR
           WHICH THE TABLE IS
           REQUIRED”; N
20 FOR     I = 1 TO 10
30 LET     M = N*I
40 PRINT   “N *” ; I; “ = ”; M
50 NEXT I
60 END
```

In this program you can get any table by giving the value of N as required by you. That is if you provide 7 with the prompt ? then you will get a table for 7, if you provide 12 then the table for 12 will be printed.

EXERCISES

ESSAY TYPE QUESTIONS

1. What is an INPUT statement ? Explain the difference between INPUT and READ statements with suitable examples.
2. You are asked to read the bio-data of 10 persons consisting of name, sex and marital status. Write a program to print the following :
   - (i) Total number of single females
   - (ii) Total number of single males
   - (iii) Total number of married females
   - (iv) Total number of married males

3. When do you use a GOTO statement and FOR-NEXT statement ? Explain with examples.
4. Explain array processing in BASIC.

SHORT ANSWER QUESTIONS

5. Write a short note on DIM statement.
6. Write a program to calculate average marks and percentage of 15 students in 3 subjects. Marks in each subject are out of 75.
7. Given the wage rate per hour and the number of hours worked, write a program to calculate total wages of an employee.
8. Write BASIC program to :
   (a) Calculate the area and perimeter of a triangle given by the formulae :

$$S = (A+B+C)2$$

$$\text{Area} = \text{SQR}\ [S*(S-A)*(S-B)*(S-C)]$$

$$\text{PERIMETER} = A+B+C$$

   (b) Calculate the volume and area of sphere using the following formulae.

$$V = 4/3\pi\ r^3$$

$$A = 4\ \pi\ r^2$$

Where r is the radius, and $\pi = 3.14$
(use INPUT statement to give value of radius)

9. Write an appropriate pair of FOR-NEXT statements for each situation described below :
   (i) A loop is to be repeated 200 times
   (ii) A loop is to be repeated from 1 to 73 with an increment of 3 :
10 Write a BASIC program to calculate the total marks for 100 students. Read Name, R.No., M1, M2 (marks in two subjects). Print out should be in following format : (HINT : Use FOR....NEXT loop to read data for one student at a time, do the computation and print the result.)

Sl. No. Name R.No. TOTAL MARKS

OBJECTIVE TYPE QUESTIONS

11. Several GOTO statements are shown below. Identify which, if any, are written incorrectly.

STATEMENTS	CORRECT/INCORRECT
(a) 100 GO TO 12	....
(b) 75 GO TO K + 1	....
(c) 30 GO TO 30	....
(d) 55 GO TO 400	....
(e) 20 GO TO "60"	....

12. Show the Output of the following program segments :

OUTPUT

(a)

```
10 FOR I = 1 TO 10
20 PRINT I, " COMPUTER"
30 NEXT I
40 END
```

(b)

```
10 FOR X = 1 TO 20 STEP 2
20 LET T = 4 + X
30 PRINT T ,
40 NEXT X
50 PRINT "END OF LOOP"
60 END
```

(c)

```
10 FOR J = 7 TO - 3 STEP - 2
20 PRINT J;
30 NEXT J
40 PRINT J
50 END
```

(d)

```
10 X = 15
20 FOR I = 1 TO X STEP 2
30 PRINT TAB (I); "*"
40 NEXT I
50 END
```

13. Several FOR statements are shown below. Identify which, if any, are written incorrectly.

CORRECT/INCORRECT

(a) 100 FOR C = .1*A TO .25*(A+B) STEP P/2 ....

(b) 65 FOR X 2 = 0 TO 100 STEP 20 ....

(c) 80 FOR K$ = P$ TO Q$ ....

(d) 30 FOR J1 = 25 TO-25 STEP -5 ....

(e) 45 FOR A = 12 TO STEP .1 ....

(f) 75 FOR X = 15 TO -5 STEP -2.5 ....

14. The skeletal structure of several FOR...NEXT loops are shown below. Identify which, if any, are witten incorrectly.

CORRECT/INCORRECT

(a)

```
10 FOR K = K1 TO K2
..........................
..........................
50 K = K+1
...........................
90 NEXT K
```

....

(b)

```
10 FOR J - 50 TO 45 STEP 2
...........................
70 NEXT J
```

....

(c)

```
10 FOR N = N1 TO N2
............................
75 ........................
............................
90 NEXT N
..................
120 GOTO 75
```

....

□ □

11

DATA PROCESSING

This chapter deals with some of the basic ideas of data processing, which is often abbreviated to EDP. Let us first try to understand the meaning of data processing and then see how computers play an important role in data processing.

Consider a retail store and purchase of some commodities by a customer at the store. Suppose that the commodities are sugar, tea, rice and potatoes. The following table provides the price per Kg and the quantity purchased :

Name of the commodity	Price per Kg	Quantity purchased
Sugar	Rs. 6.50	2
Tea	Rs. 50.00	1
Rice	Rs. 5.00	10
Potatoes	Rs. 2.00	2

The cost of each commodity is found by multiplying Price perKgwith quantity purchased and total cost is found by adding the costs of all items. Thus, the transaction of purchase is described by the data containing

(*i*) Commodity name
(*ii*) Price per Kg
(*iii*) Quantity of each commodity purchased.

Processing of data consists of :

(*i*) multiplication
and (*ii*) addition.

Information obtained after processing data is the total cost to be paid by the customer. Thus "data" is used during "Processing" to produce "Information". This complete process is called "Data Processing". Now let us consider the example of preparing results of a class. Suppose a teacher conducts three class tests and a final examination. At the end of the year, the marks must be processed and final mark sheet with percentage of marks is to be submitted to the Principal of the school. The final mark sheet should contain the name and roll number of each student along with the marks in the three tests, marks in final examination, total marks and percentage of marks.

Total marks for each student are obtained by adding marks of the three tests and final examination marks and the total maximum marks are obtained by adding the maximum marks of the three tests and final examination. Now the percentage of marks are obtained by multiplying total marks by 100 and dividing the result by total maximum marks.

When data is processed on a computer, it is often organized as records. A record is a collection of related data items or fields. Each record normally corresponds to a specific unit of information. For example, the record of each student is described by the data containing :

(*i*) Name
(*ii*) Roll number
(*iii*) Marks in each of the three tests
and (*iv*) Marks in final examination.

The collection of records is called a file. A file contains all of the related records for a problem. Therefore, in the result preparing example, the file contains all of the records required to announce the result. Files are stored on magnetic tape or magnetic disc.

Processing of data consists of addition, multiplication and division. The information obtained after processing of data is the total marks and the percentage of marks for each student.

Now let us define the words Data and Data Processing more precisely.

DATA

The word 'datum' of which the word 'data' is plural, means a fact or figure or more specifically, a value of an attribute of an entity. An entity, in general, may be an object, idea, event, condition or situation. Thus, a student, a purchase-event are entities. A set of attributes describes an entity. The values of these attributes are data about the entity. In the above example, student is an entity and the attributes of the entity that we have used are Name, Roll number, Marks in three tests separately and Marks in the final examination. Thus we are using 6 attributes of the entity "student". The values of these attributes are called "data".

Examples of data are :

Name	:	Ramesh, Mohan
Roll No.	:	1085, 1296
Marks	:	85, 54, 92, 47, 63

Usually, we select only those attributes of entity which are relevant to us. For example, the colour of the commodity purchased by the customer is not important in calculation of the cost. Hence, although the colour of the commodity is the property of the commodity, it is not relevant here. Similarly, for paying the salary to an employee, his height is not relevant. On the other hand, his name and salary are relevant.

DATA PROCESSING

Data by themselves are not sufficient to solve a problem. By organizing, calculating and transforming data we arrive at a solution. Data have no informational content or problem solving ability by themselves. When processed to solve a problem, data becomes information.

The term 'data processing' implies transformation of data into more useful form. Data processing includes not only numerical calculations, but also operations such as organization of data, classification of data, sorting of data, transmission of data from one place to another etc.

INFORMATION

The term information to a layman means processed data. The results obtained by data processing are called **"information"**. However information produced in one data processing step may be used as data in the next data processing step.

In the example above, results for one subject of a class are submitted to the Principal of the school. Assume that such results are to be prepared for each subject of the class and are to be submitted to the principal of the school. Results of the each of the subjects submitted to the principal are further processed in order to prepare complete result of the class. Processing of complete results require finding the total marks for each of the subjects and then finding the percentage of total marks. If the percentage of total marks is 60 or more then grade 'A' is assigned to the student. If the percentage of the total marks is 50 or more but less than 60 then grade 'B' is assigned to the student. If the percentage of total marks is less than 50 or the percentage of marks in any subject is less than 50 then grade 'F', that is failed, is assigned to the student. When the complete results are prepared it would give the following information ;

(*i*) the number of students getting grade 'A'
(*ii*) the number of students getting grade 'B'
(*iii*) the number of students failed.

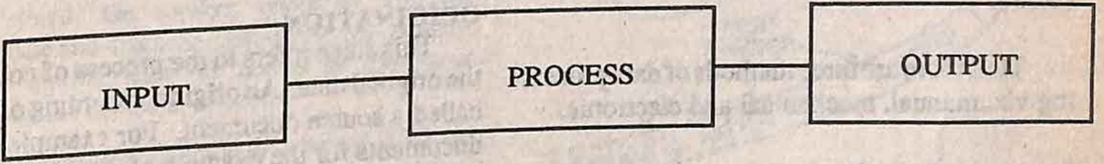
In the above example, you must have noticed that the information produced in the previous example, that is the results prepared for each of the subjects, is the data for this example. This shows that the information for one person may be the data for another person or *vice versa*. Hence it is very difficult to distinguish between data and information, which is dependent on the context. The only difference between the data and information we can make out is 'what exists' and 'what is required'.

From the above discussion it is evident that the processing consist of one or more of the following activities.

- Calculation (e.g.,finding percentage of marks)
- Comparison (e.g.,comparing percentage with 60 or 50 to decide grade)
- Logic (eg. carrying out arithmetic operation in the correct sequence)
- Decision (eg. declaring result)

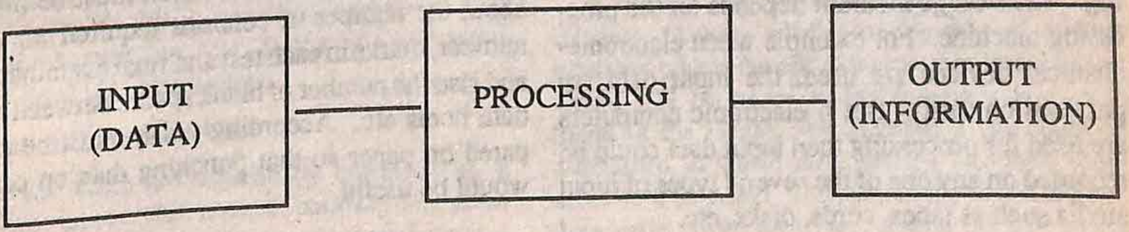
SYSTEM

A system is defined as a group of inter-related components that seeks to attain a common goal by accepting inputs and processing them into output in an organized process. Thus, a production system accepts raw materials as inputs and produces finished goods as output with a goal to maximize profit. The components of this system are men, machines and material, which interact with one another in an organized manner.



SYSTEM

A data processing system can be viewed as a system that accepts data as input, processes it into information as output.



The data processing system is called **manual** when the processing is performed by human beings. In the manual data processing system, the components are human beings, that is, the clerical staff who perform processing to produce information. The tools used by them for processing would be paper, pen, carbon papers, pencils, files, slide rules and abacus etc.

Data processing can also be performed using 'mechanical devices' such as typewriters, calculators, book keeping machines, posting machines and duplicating equipment etc.

Manual and mechanical data processing is called **non–automatic data processing.** Data processing performed using electro-mechanical devices, computers, card readers, keypunch, paper tape punch etc.,is called automatic data processing or electronic data processing (EDP).

Thus there are three methods of data processing viz.,manual, mechanical and electronic.

Stages in Data Processing

As explained above data processing consists of the following three basic stages :

(*i*) Input
(*ii*) Processing
(*iii*) Output

Input

In this stage initial data are stored in some convenient form on some medium for processing. The storage medium depends on the processing machine. For example when electromechanical devices are used, the input data are punched on cards. But if electronic computers are used for processing then input data could be recorded on any one of the several types of input media such as tapes, cords, disks, etc.

Processing

In this stage the input data are changed and usually combined with other information to produce it in a more useful form. Thus, a summary of sales for the month can be obtained from the sales order.

Output

In output the results of the processing step are collected. The particular form of the output depends on the use of the output. For example, output data may be pay cheques for the employees, a printed summary of monthly sales for management, or simply data to be stored for further data processing at a later date or bar charts. pie charts etc., describing various information.

Frequently, three more stages are added to the basic data processing. These are origination, preparation and storage as shown below :

ORIGINATION

This stage refers to the process of collecting the original data. An original recording of data is called a source document. For example source documents for the example of preparing results are the graded test papers and examination papers of the students. The source of the data may not be the same, that is the data required to get information may originate at different sources.

PREPARATION

The collected data is transformed into some convenient form for input. For example, when punched cards are to be used for input in the result preparing example, data must be prepared appropriately. This means a decision must be made about the number of columns required for roll number, marks in each test and final examination and also the number of blank spaces between two data items etc. Accordingly data must be prepared on paper so that punching data on cards would be useful.

STORAGE

Data processing results are frequently placed in storage for further use as input for processing at a later date.

One of the major requirements in order to utilize computer systems in all areas of application is the ability to store and access data. On small computer systems, the amount of data stored and accessed might not exceed one or two million characters. On large computer systems, billions of characters of data might be stored and accessed. Large utility companies, for example, may require that all customer records be stored and be accessible for inquiries about payments and billing.

Thus, auxiliary storage (sometimes called secondary storage) is an extremely important part of the data processing installation; for without access to data, processing could not occur.

There are two major considerations when examining the auxiliary storage used with computer systems. The first is the type of media on which the data is stored. On modern computer systems, magnetic tape and magnetic disk are the two primary means of storing data for access by programs executing on the computer system.

The second consideration is the manner in which the data is organized and accessed. A proper choice of the organization and access method can enable an application to access and process data with maximum efficiency.

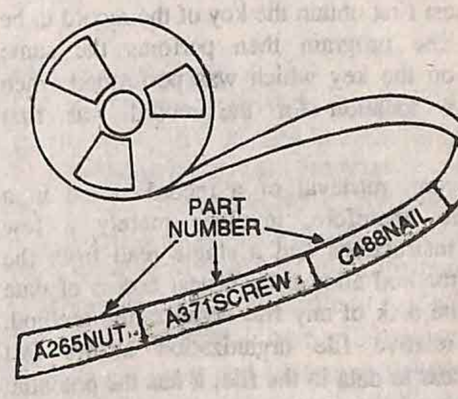
We have already discussed the media on which data is stored in Chapter 5. We will now discuss the manner in which data is organised and accessed.

When data is stored on auxiliary storage, it must be logically organized into files. There are three major file organization methods which are used for storing data on auxiliary storage: 1) Sequential organization; 2) Relative or direct organization; 3) Indexed organization. The following paragraphs discuss these file organizations.

Sequential file organization

Records which are sequentially organized in a file are stored one after another, normally in a given sequence. The sequence is usually based upon a control field or key in the record. Records which are organaized sequentially are also retrieved sequentially for processing.

Sequential organization must be used when magnetic tape is the storage medium and can also be used with disk. This means of organization is used most often in batch processing systems where input records can be batched and sorted in the same sequence as the records stored in the file on auxiliary storage. These files are then processed against one another. Since sequential files require sequential accessing, they are not used too often in transaction-oriented systems because of the limitation on rapidly accessing records.



The inventory file above contains records stored sequentially. The records are stored in an ascending sequence by part number. Since these records are stored sequentially, they will also be accessed sequentially.

Relative or direct file organization

A relative file (sometimes called a direct file) contains records which are stored in relative locations within the file. The location into which a record will be stored is based upon a key value found in the record. For example, a file could be established in which there are ten locations where data could be stored. If the key in the record is a one digit value (0-9), then the value in the key would specify the location within the file where the record with the key 3 would be placed in relative location 3: the record with the key 6 would be placed in relative location 6; and the record with key 7 would be placed in relative location 7.

Once a record is stored in its relative location within a relative file, it can be retrieved either sequentially or randomly. Sequential retrieval can occur by indicating that the record from the first relative location is to be retrieved, followed by the record from the second relative location, and so on. Thus, all of the records stored in the file would be sequentially retrieved based upon their relative location within the file.

The more common access method for relative files, however, is random access. In order to randomly access a record stored in a relative file, the program must first obtain the key of the record to be retrieved. The program then perfoms the same arithmetic on the key which was performed when the relative location for the record was first determined.

The random retrieval of a record stored in a relative file, therefore, involves merely a few arithmetic instructions and a single read from the disk. This method allows the fastest access of data stored on the disk of any file organization method. Although relative file organization allows fast random access to data in the file, it has the potential for wasting a great deal of storage space on the disk.

Indexed file organization

The third major file organization method is indexed file organization. Records are stored in an indexed file in an ascending or descending sequence based upon the key of the record. In this respect, the records stored in the file resemble records stored in a sequential file.

An indexed file, however, also contains an index. An index consists of entries containing the key to the data record stored in the file and also the disk address of the record in the file. This index is usually stored in the file on disk when the file is created and is retrieved from the disk when a record in the file is to be accessed.

Records can be accessed in an indexed file sequentially or randomly. Sequential access can occur in onr of two ways. First, sequential retrieval of records in the file can begin with the first record in the file and continue through the file. Another technique for sequentially retrieving records is to begin the retrieval with a record with a given key.

Random access can also be used with indexed files. To randomly access a record in an indexed file, the index is searched until the key of the record to be retrieved is found. The address of the record found in the index is then used to access the record directly from the file without reading any other records.

Even though both relative and indexed files offer sequential and random access, there are several distinct differences. Random retrieval when using relative files is faster than when using indexed files because an index does not have to be searched when using relative files. Instead, the relative record location is calculated and the record is read. Searching the index can be time-consuming for two reasons: 1) The index may be quite long and the search, even at main computer storage speeds, may take some time; 2) Since indexes are often stored on auxiliary storage and are not kept in main computer storage, random retrieval of an indexed file may require that the index is read into main computer storage before it can be searched for the matching record key. Thus, a minimum of two disk reads—one for the index and one for the desired record—may be required. Each time records must be read from the disk, the retrieval operation is slowed.

The difference when sequentially retrieving data from an indexed file and a relative file is that the data sequentially retrieved from the indexed file will be in ascendiang or descending key sequence because the records are stored in the file in this sequence. With relative files, on the other hand, this is not always true.

Indexed files offer a compromise between sequential files and relative files. In many applications, they prove to be ideal because they allow both random retrieval and sequential retrieval in key sequence.

METHODS OF DATA PROCESSING

Consider an example of business organization where information is used for day-to-day administration and policy decisions. As the complexity of business increases, the number of functions to be performed and resources to be controlled in an organization also increase and consequently the complexity of data processing also increases. The necessary data processing may be carried out manually or mechanically or electronically. The organization operates continuously and as such continuously monitored. The data processing system must be responsible to supply information when it is needed, so as to make the performance of the organization optimum. The data processing system is not considered satisfactory if it fails to supply information in time. The manual methods often fail due to long processing time, inconsistency and unreliability. To meet the needs of an organization successfully, it is necessary to use computers. A computer has the advantage of speed, consistency and reliability over human beings. The above example illustrates under what circumstances computers should be used for data processing. From the above discussion, we conclude that methods of data processing depend on processing requirements and processing requirements are determined by

- (*i*) Volume of data
- (*ii*) Complexity of data processing operations
- (*iii*) Processing time constraints
- (*iv*) Computational demands.

The question of when does an organization select one method of data processing over another is largely an economic decision and the above listed processing requirements.

Now let us examine various types of Data processing. There are three types of data processing systems :

- (*i*) Batch processing
- (*ii*) On-line (Real-time) processing
- (*iii*) Distributed data processing

Batch Processing

One of the traditional task for the computer has been payroll. In this the system collects data over a period of time such as weekly, monthly etc. Once the program of instructions has been devised, it can be used on an indefinite number of employees. This kind of processing is known as Batch-processing. That is in Batch processing, the information is collected over a period of time-daily, weekly or monthly, and the processing is done on the batch as a whole and repeated the following day, week or month. Many computer applications currently running are Batch processing applications.

On-Line Processing

The data processing system where data are processed as soon as it occurs is called on-line system. A reservation system for train or airline is an example of on-line system. As soon as a customer makes a request for a seat, the request is immediately processed and the information, that is whether the seat is available or not, will be provided to the customer. There will be a considerable loss of business and customer dissatisfaction if we ask them to wait for the result of their request until the end of batch period. So for this type of applications we must use only on-line systems.

Distributed Processing

Some businesses such as banks, Airlines etc. are distributed over a wide area, and each one requires access to common business information. To solve this, a computer will be kept centrally with all the information and each of the offices are directly connected to this centrally located computer. When a request is made to the Central Computer through the terminal provided at a branch, the central computer processes the data and provides information at the branch. This type of processing is known as **distributed data processing.**

EXERCISES

ESSAY TYPE QUESTIONS

1. Explain with examples the difference between data and information.
2. Illustrate with an example typical steps in a business data processing problem.
3. What are the steps involved in data processing activity? Explain each step in 3 to 4 lines.
4. Describe the terms : **Transaction** and **Batch** in the context of Inventory Processing.

HORT ANSWER QUESTIONS

5. Define the terms :
   (*a*) Data Elements
   (*b*) Records
   (*c*) Files
6. What is a system? Describe in 30 to 40 words.
7. What are the various methods of data processing ? Explain each in 2 to 3 lines.
8. How is a data prepared ?

OBJECTIVE TYPE QUESTIONS

9. Fill in the blanks :

(*i*) A set of related data items is called a ....

(*ii*) A set of similar records is called a ....

(*iii*) .... and .... are two kinds of data processing.

(*iv*) The full form of EDP is ....

(*v*) The word .... means facts, and the word .... means working with facts.

(*vi*) A .... is defined as a group of interrelated components that seems to attain a common goal by accepting inputs and processing them into output in an organized way.

(*vii*) A data processing system is called .... when the processing is performed by human beings.

(*viii*) The other name of automatic data processing is ....

□□

S.C.E.R.T., W.B.
Library